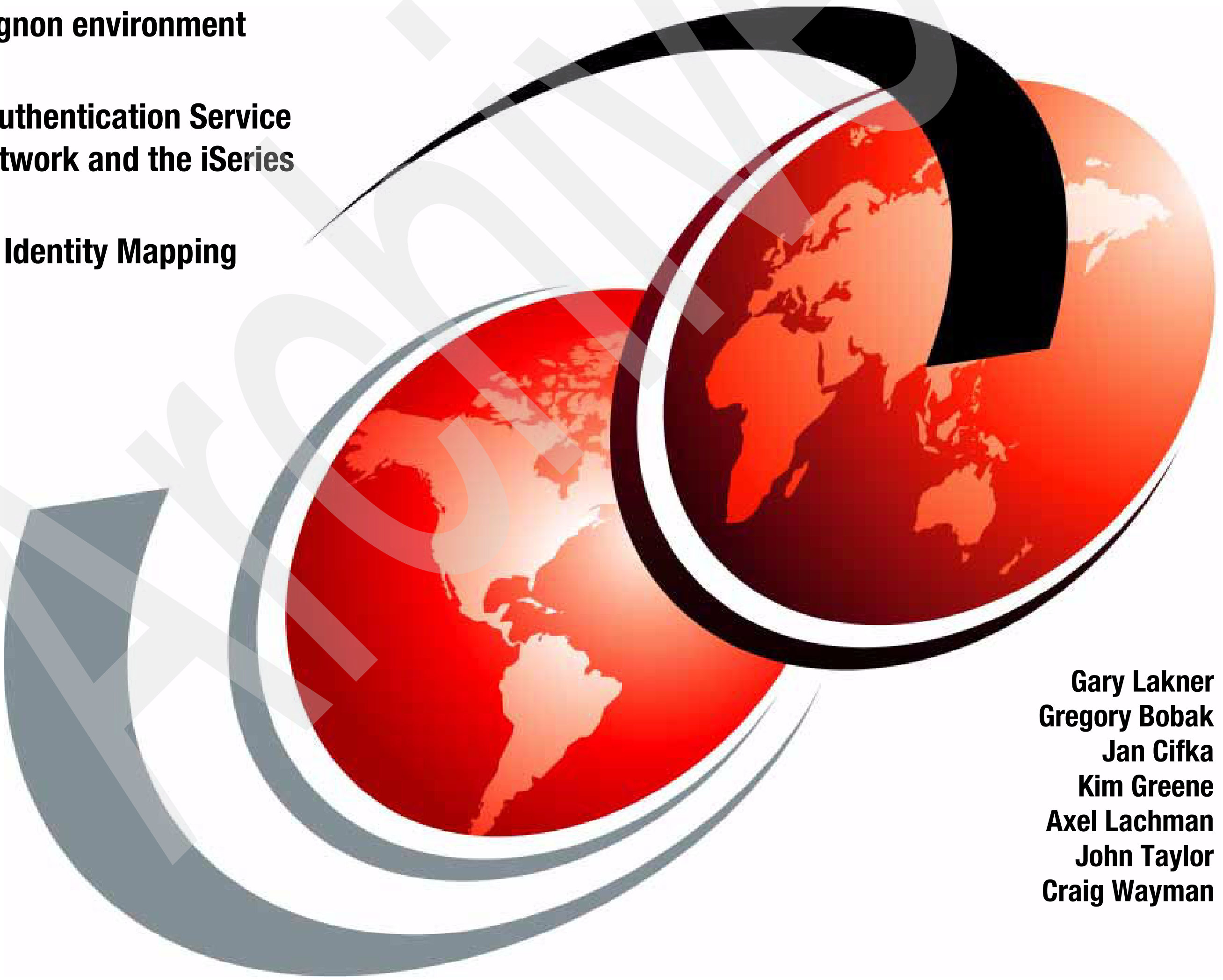
Windows-based Single Signon and the EIM Framework
on the IBM eserver iSeries Server

Configure your iSeries to participate in a single signon environment

Network Authentication Service on your network and the iSeries

Enterprise Identity Mapping explained



Gary Lakner
Gregory Bobak
Jan Cifka
Kim Greene
Axel Lachman
John Taylor
Craig Wayman

ibm.com/redbooks

Redbooks

International Technical Support Organization

Windows-based Single Signon and the EIM Framework on the IBM @server iSeries Server

April 2004

SG24-6975-00

First Edition (April 2004)

This edition applies to Version 5, Release 2, Modification 0 of OS/400.

Contents

Notices

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

eserver™	AIX®	Notes®
eserver™	AS/400®	OS/400®
e-business on demand™	Distributed Relational Database	PartnerWorld®
ibm.com®	Architecture™	Redbooks™
iNotes™	Domino®	Redbooks(logo) ™
iSeries™	DB2®	RACF®
pSeries®	DRDA®	RS/6000®
xSeries®	IBM®	S/390®
z/OS®	Lotus Notes®	SecureWay®
zSeries®	Lotus®	WebSphere®

The following terms are trademarks of other companies:

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Preface

Support for a Kerberos based Network Authentication Service and the introduction of Enterprise Identity Mapping (EIM) were two exciting OS/400® V5R2 announcements during 2002.

A Kerberos based Network Authentication Service enables the IBM® @server™ iSeries™ (and any enabled application) to use a Kerberos ticket for authentication instead of a user ID and password. Enterprise Identity Mapping is a cross platform solution that involves a wide range of technologies including Kerberos, LDAP, and Kerberos Network Authentication Service. Together they provide an extended single signon solution.

This IBM Redbook is all about that extended single signon (SSO) solution. Our goal is to help you plan, understand, implement, and troubleshoot Enterprise Identity Mapping (EIM) and Network Authentication Service. We make the assumption that you already have Kerberos authentication on your network, or at least have the pieces available to implement a basic authentication scheme using Kerberos. Our scenarios all use a Windows® 2000 Key Distribution Center (KDC) but that service can be provided by any platform that supports the KDC functions.

In Part 1 we give a high level overview of SSO and things to consider before implementation. In Chapter 1 we explain, in layman's terms, what are the key components of an SSO solution, and we outline the unique IBM @server approach to SSO using Enterprise Identity Mapping. This is intended for a general audience. Chapter 2 explains the network and software requirements required to implement a more complete single signon solution including Enterprise Identity Mapping. This will be of interest to administrators. Chapter 3 describes the scenario on which we base our examples.

In Part 2 we give a more detailed technical explanation of the three components used in our single signon solution: Kerberos, Network Authentication Service on the iSeries, and EIM. This will be of interest to administrators and developers who need to enhance their technical knowledge and learn about the capabilities of both Network Authentication Service and EIM.

In Part 3 we give detailed step-by-step instructions showing how to configure the SSO environment with EIM. This implementation section is complete with additional scenarios and a discussion of the Java™ programming APIs. Part 3 will be of interest to administrators and developers.

In Part 4 we cover troubleshooting of the setup and operation of the environment. This is intended for administrators.

There are several key points to keep in mind while reading this book. These ideas will be further discussed in several of the chapters.

- Single Signon and Enterprise Identity Mapping are not the same thing. They work together to create a complete environment.
- The creation of this environment does not have to be an "all or none" event. It can be implemented in incremental steps if that suits your requirements.
- Authentication and authorization are not the same thing.
- Although this book focuses on a Windows based authentication model, it is not the only platform that is capable of providing a Key Distribution Center (KDC).
- Kerberos is not the only form of authentication that can be used to implement EIM.

This book is primarily intended for three audiences:

- Business people who need to understand the costs, benefits and implications of SSO.
- Administrators who are to implement EIM-based SSO solutions.
- Developers who want to take advantage of EIM to enhance their applications.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Rochester Center.

Gary Lakner is a Staff Software Engineer in the IBM Rochester iSeries Support Center. He has six years of experience working with the iSeries in the area of TCP/IP and TCP related applications.

Gregory Bobak is Senior Programmer Analyst for Cole National in Twinsburg, Ohio, USA. He has been with Cole National for four years and has six years experience working on the iSeries. He and the e-Business team at Cole National are nationally recognized for their development and use of Web services. This is the second redbook he has co-authored. He has a bachelor's degree in Computer Science from Bowling Green State University.

Jan Cifka is a Senior System Engineer at Trask Solutions, an IBM BP in Prague, Czech Republic, and president of Common Czech Republic. He holds a degree in Theoretical Physics from Charles University in Prague. He is an IBM @server Certified Specialist iSeries V5R2, both in System Administration and Technical Solutions Design. He has more than 30 years of experience in IT with IBM AS/400® and iSeries systems since 1990 and compatible mainframes earlier. For many years he has been highly active in IT education, recently speaking about iSeries security at Common Europe and Common Czech Republic conferences.

Kim Greene, President of Kim Greene Consulting, Inc., specializes in Domino® for iSeries consulting, development, customized education, and performance. Kim has over six years of experience with Domino and 14 years of experience with the AS/400 and iSeries platforms. Kim specializes in iSeries Domino performance analysis, system and application tuning, enterprise integration with back-end applications and data, and Domino application development. Other services offered include providing customized iSeries Domino education and training for customers and Business Partners. For more information about any of these services, visit her Web site at www.kimgreene.com. Kim is also a frequent writer for technology magazines and speaks at many conferences worldwide. Previously she was employed at IBM where she worked in the PartnerWorld® for Developers (PWD) organization helping IBM Business Partners incorporate Domino into their existing applications. Prior to working in PWD she worked in several areas of OS/400 performance in the AS/400 development laboratory.

Axel Lachmann is a Project Manager and Senior Systems Engineer at FoxCom, an iSeries Business Partner in Germany. He has 11 years of experience in the OS/400 field. He is an IBM Certified Solutions Expert - AS/400 Technical Solutions. His areas of expertise include e-business enablement of LOB applications, application modernization, Server Consolidation with LPAR, Windows integration and Linux planning and implementation. Axel also teaches extensively about e-business related topics and technical certification courses for IBM Learning Services in Germany. You can reach Axel at alachmann@foxcom.de.

John Taylor is Technical Director of the Typex Group and a Board member of COMMON in the UK. He holds a degree in Civil Engineering from Leeds University, England. He has 20 years of experience in the fields of IBM midrange servers and groupware, specifically IBM iSeries, Lotus® Domino and LDAP directories. He is the architect of the BlueNotes family of software products as marketed by IBM. For more details, visit www.bluenotes.com. He has written extensively on these topics and is a regular speaker at international technical conferences.

Craig Wayman is a Senior Domino Developer for the Typex Group in the UK. Craig has recently graduated from the University Of Salford, England obtaining a BSc. Hons. in Computer Science. While on placement from the Computer Science course he worked at IBM Lotus writing Domino and Java applications. Applications which are used worldwide by IBM and IBM Lotus sales teams to provide services for customers wanting to purchase products and services. Since then Craig has expanded into the iSeries platform continuing to provide innovative Domino applications and solutions to meet the needs of his organization. Currently Craig is working on a suite of tools to help administrators manage an organization's EIM environment.

Thanks to the following people for their contributions to this project:

Jim Cook
Deb Landon
Brian Smith
International Technical Support Organization, Rochester Center

Pat Botz
Therese Dalton
Steven Hansen
Brian Krings
John McMeeking
IBM Rochester

Alan Cohen
IBM Raleigh

We extend a special thank you to the following contributors:

- Ted E. Pshock of Cole National for creating and submitting the RPG programs used in the WebSphere® scenarios section.
- Craig Robbins, who submitted the iNotes™ scenario.
- Anna G. O'Neal for submitting the Host On-Demand scenario.
- Thomas Barlen for submitting his EIM demo tools.

We also thank the many people who reviewed this book in its draft stages and provided feedback.

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  `ibm.com/redbooks`

- Send your comments in an Internet note to:

  `redbook@us.ibm.com`

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. JLU Building 107-2
  3605 Highway 52N
  Rochester, Minnesota 55901-7829

Part 1

Introduction to single signon and Enterprise Identity Mapping

Part 1 is dedicated to giving you an overview of, and a feel for, our implementation of single signon (SSO). We explain why businesses and network administrators would want to implement SSO. We take a look at the components of what we believe is a more complete SSO solution and some of the possible savings that may be realized by implementing our single signon strategy. After examining the business aspects of using SSO we provide a planning chapter. Included are the minimum software requirements and planning forms that will help your implementation go more smoothly. We then describe our scenario, on which we base our step-by-step instructions.

This part is intended for business people, administrators and developers.

1

An overview of single signon

In this chapter we explain the advantages of implementing single signon (SSO). Next, we provide a high level overview of the components of a single signon solution and of how they work together. Then, we describe the unique IBM @server approach to single signon using Enterprise Identity Mapping (EIM).

1.1 Why single signon?

If you look at the prospect of implementing single signon in your network environment you must weigh the costs of the implementation against the benefits of the end results. We are confident that you will be able to see positive gains from each of the perspectives examined in this section. Users will have less down time organizing and managing passwords. Ongoing administration costs will be reduced because the overhead of password related help desk calls and the number of user registries could be decreased. An additional benefit will be added security on your network. From the business view, you end up with a more secure network that costs less to maintain. Since the implementation of our complete single signon solution is not an “all or none” venture you control how much you want to invest implementing it and when.

1.1.1 What is single signon?

In a typical environment today most employees have multiple user profiles and passwords (Figure 1-1) to remember. Each time they try to access a system they must supply their ID and password to access the resources on that system. A simplified explanation of single signon (SSO) would be that single signon offers the prospect of a computer user signing on once at the start of the day and not be prompted for any additional signons or challenged for additional passwords the rest of the day. In a perfect world that would mean that the user could securely access all the resources that they require during the course of that day without having to go through another login process.

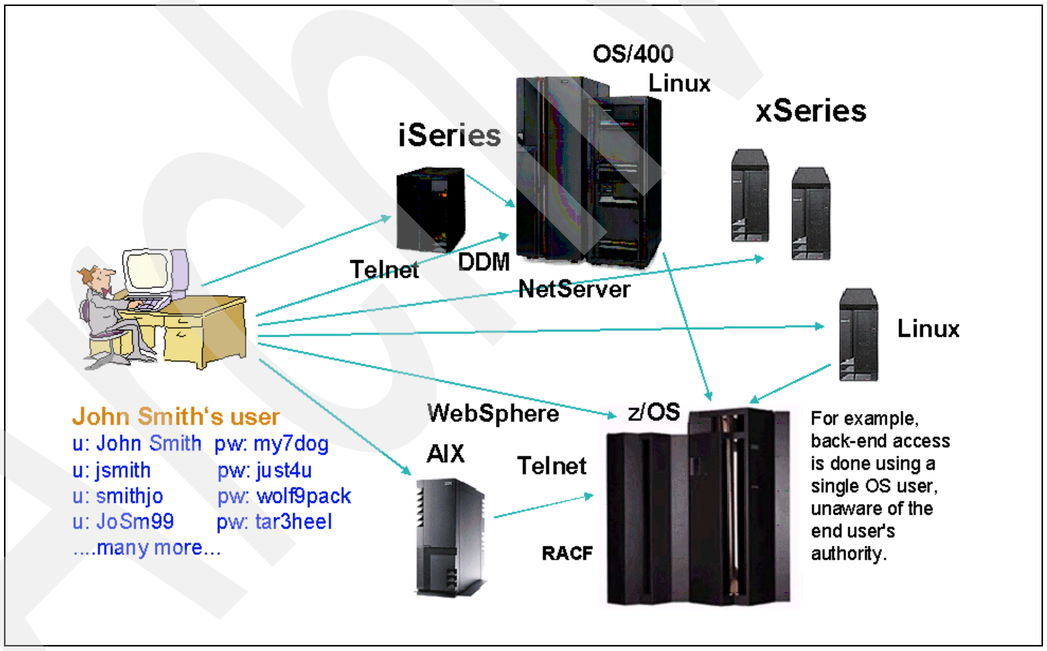


Figure 1-1 Typical network environment today

Well, we haven’t quite reached that “perfect” status yet but we are close enough to make the implementation of SSO worthwhile. Section 1.5, “Currently enabled iSeries applications” on page 13 provides a list of iSeries applications that are SSO enabled.

1.1.2 What are the benefits of single signon?

Single signon can solve many problems across the enterprise. We can consider these problems from three different perspectives - those of the user, of the developer and of the organization.

The problem for the user

In a typical client-server environment today the average user signs on to many services - workstations, servers and applications. For each of these services there may be separate IDs and passwords. Some of the services accessed in this way will have rules about how the ID is defined, such as upper or lower case and whether special characters are allowed. Not all services will have the same ID rules. Some services will have rules about how to define the password, such as a minimum length or the inclusion of numeric digits. Not all services will have the same password rules. These same services may also insist on the passwords being changed at regular intervals. Not all services will have the same password change intervals. After a short while the user will be required to keep track of numerous IDs and passwords. Not only is memorizing, synchronizing, and maintaining all of the IDs and passwords frustrating for the user, it costs that employee valuable time.

The problem for the developer

Developers of multi-tier client-server applications usually find that they have to authenticate the user to the second tier of an application, for example, a Web server, and then have a single common ID used for all requests to the back-end business system. This approach does not easily allow for the data retrieved to be varied according to the user. It may also mean that this single ID and password have to be hard-coded into the application. It could also require that this high-level ID and password flow over the network in a detectable form.

An alternative approach, because each tier maintains its own user registry, would be to find a way to map user identities from one tier of an application to another. Developers often do this themselves by creating yet another registry containing the mapping between the identities. This increases the cost of developing the application and of administering the IT environments that deploy these applications.

The problems for the organization

The problems for the organization flow largely from those for the user and the developer. There is a loss of user productivity in managing and changing passwords. It simply takes up a lot of business time. One individual may not spend a high percentage of their time on this task but the total time lost can be considerable when spread over the entire organization.

Help desk costs

There is also the more measurable loss of time for help desk staff who are involved in the resetting of passwords. In a report issued by Gartner, it was estimated that 20% to 40% of all calls to help desks involve forgotten passwords costing companies $14 to $26 per reset. The report can be found at:

`http://www3.gartner.com/Init`

Administrative costs

In addition to helping users with their problems the system administrators are often involved in maintaining the new user registries which the developers have created. They also must find appropriate mechanisms for tracking information about a person and their user identities when they join or leave the organization or change roles.

Security exposures

The user's reaction to the need to remember so many passwords can be twofold - to make them simple to remember or to write them down. Either method renders them open to unauthorized detection and use.

In addition, the need to transmit passwords over the network can lead to their detection via network sniffing techniques.

The related problem of a lack of a single point of user signon management leads to a further security exposure. Staff leaving the organization can remain enrolled in a variety of services and their IDs and passwords may be used by them or by others after their departure. This same lack of a single point of user management can also lead to the organization paying for too many software licences because the number of enrolled users is artificially high.

1.2 Vertical versus horizontal SSO

To better explain single sign on we have devised two conceptual models of single signon, vertical and horizontal. This section describes the differences between the vertical and horizontal models of SSO. It addresses many misconceptions about SSO, especially the idea that single signon, once implemented, will automatically eliminate the need to sign on to other systems or applications.

1.2.1 Vertical SSO

Vertical SSO is perhaps the most widely understood model for SSO. It is the idea that once a user signs on to a network, the user is automatically signed on to all applications, systems, and resources that support SSO on the network. Typically, however, not all applications, systems, and resources are SSO enabled, nor is it always practical to have all components enabled for SSO. In Figure 1-2, the user signs on to all applications and systems that are SSO enabled. However, the x client is not SSO enabled and is outside the vertical stack. Consequently, the user must explicitly sign on through the client to connect the attached system. With proper implementation of vertical SSO, users and administrators are required to maintain fewer passwords, allowing both to become more productive by reducing the number of calls to the help desk.

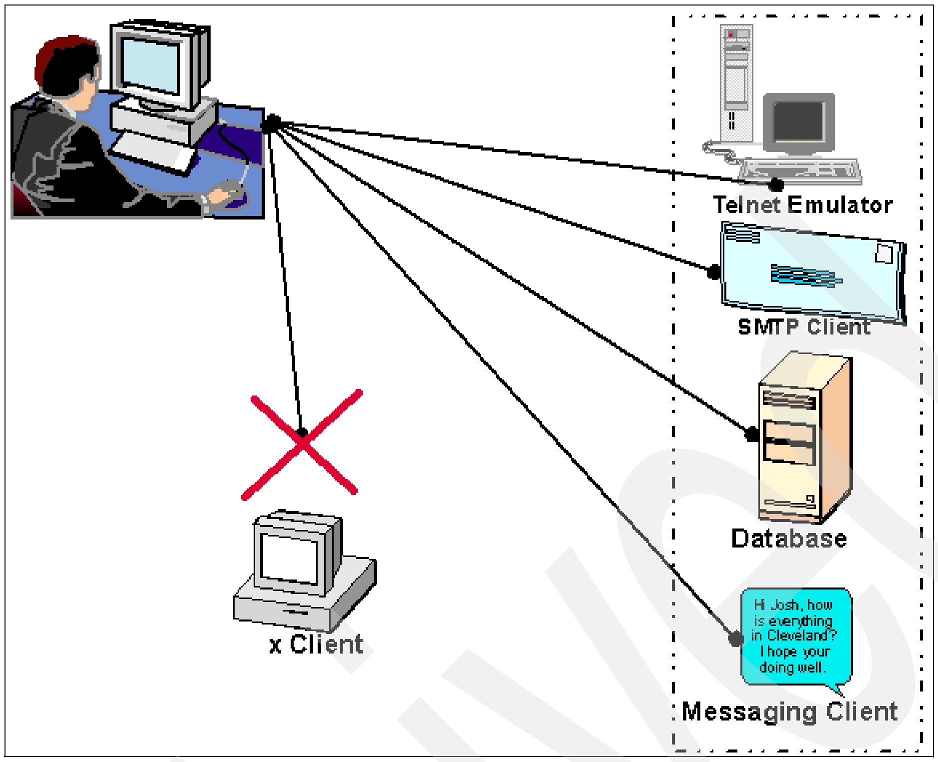

Telnet Emulator
SMTP Client
Database
x Client
Hi Josh, how is everything in Cleveland? I hope your doing well.
Messaging Client

Figure 1-2 A vertical signon stack

1.2.2 Horizontal SSO

Horizontal SSO is the concept of connecting to multiple applications that work together over a multiple tier environment using only one signon. In Figure 1-3, the user signs on to the network which automatically signs him on to the telnet emulator. From the telnet emulator the user accesses a SSO enabled Enterprise Information System (EIS) application on the remote server that hosts the telnet application. Another explicit signon is not required. The EIS application then uses SSO to connect and sign on to a remote database server to retrieve requested data. Again no explicit signon is required.

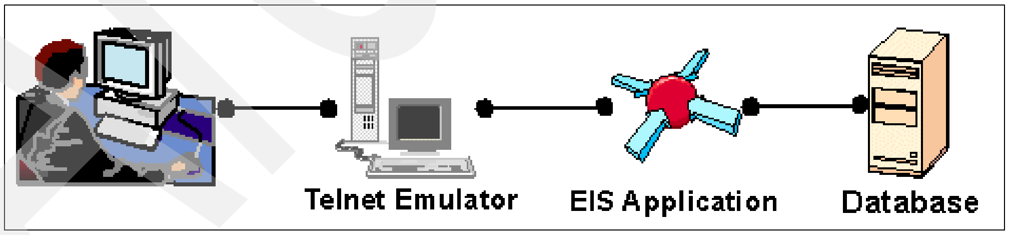


Figure 1-3 Horizontal SSO

Horizontal SSO, used with a consistent, pluggable framework, can increase productivity of both developers administrators. Developers benefit from a consistent, reusable method of authenticating across a multiple tier environment. Administrators may implement horizontal SSO in a manner that requires no passwords for users on each tier. This reduces user maintenance, and increases security by ensuring certain systems and applications are only accessed through pre-defined applications and methods.

1.2.3 Vertical and horizontal signon work together

Perhaps the most common approach to SSO in the enterprise will be a hybrid of both vertical and horizontal SSO. In Figure 1-4, it is pretty clear that SSO serves more pieces of the enterprise when both vertical and horizontal SSO are implemented. Although this combination is more pervasive than either a vertical or horizontal strategy alone, it is nearly impossible to touch all applications, systems, and resources.

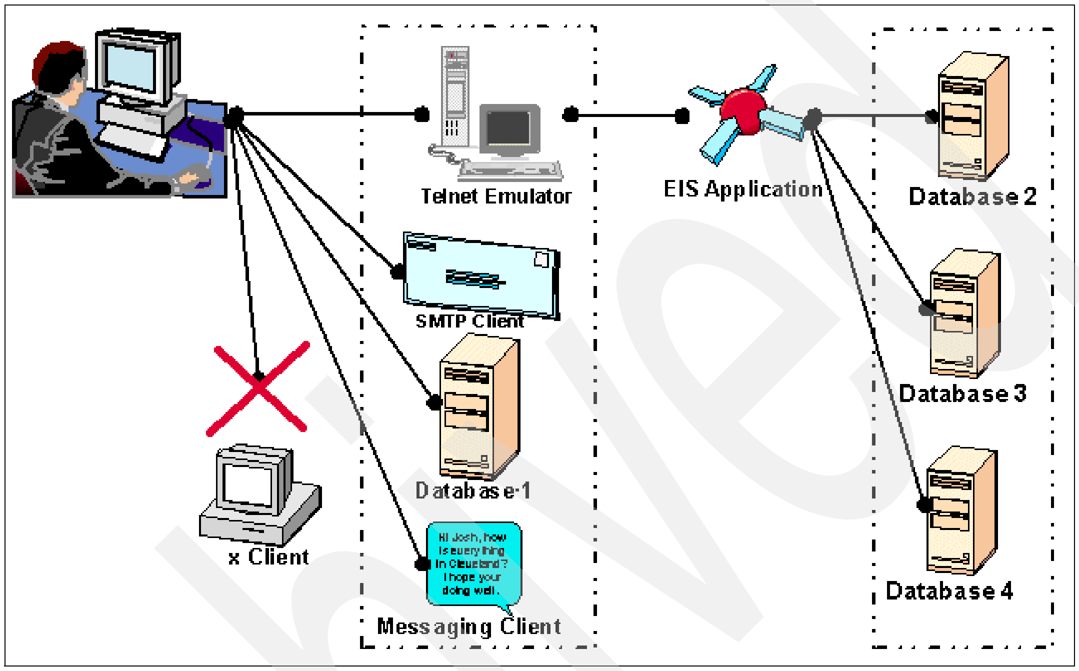


Figure 1-4 Vertical and horizontal SSO working together

1.3 How SSO works

Single signon (both vertical and horizontal) can be achieved in a variety of ways. In this section we explore SSO methodologies that comply with basic security fundamentals and reduce administrative overhead.

1.3.1 Authentication, authorization and auditing

Authentication, authorization and auditing are the three basic components of any security scheme.

- Authentication concerns how we verify the identity of a user or entity. Traditional authentication utilizes a name and a fixed password. Authentication generally takes place when the user first logs in to a machine (or domain) or requests a service from that machine or the network.
- Authorization concerns the granting of the appropriate access rights to data and applications. It is usually described as granting a principal the rights to an object. Generally, although not necessarily required, authentication precedes authorization. It would seem that authorization itself is meaningless without proper authentication. In the past, authentication and authorization have been tightly coupled in that the same mechanism may be used for both. As we will see below, these can now be separated.

- Auditing is the action of recording what a user is doing, and/or has done and store them for future reference. In the scope of this book, auditing can be used to look for suspected unauthorized access attempts.

1.3.2 What is Kerberos?

Kerberos may be described as a secure, trusted, mutual authentication service. Kerberos maintains a central database of all identities - including users, computers and services - in a realm. It works on the basis of securely authenticating a user, usually by password, and then issuing secure 'tickets' to both users and applications who exchange them with each other. This means that an application can trust Kerberos to have authenticated the user and hence does not need to authenticate them a second time. The user then accesses the application without further signon. The user can also be sure that the application is the one it appears to be.

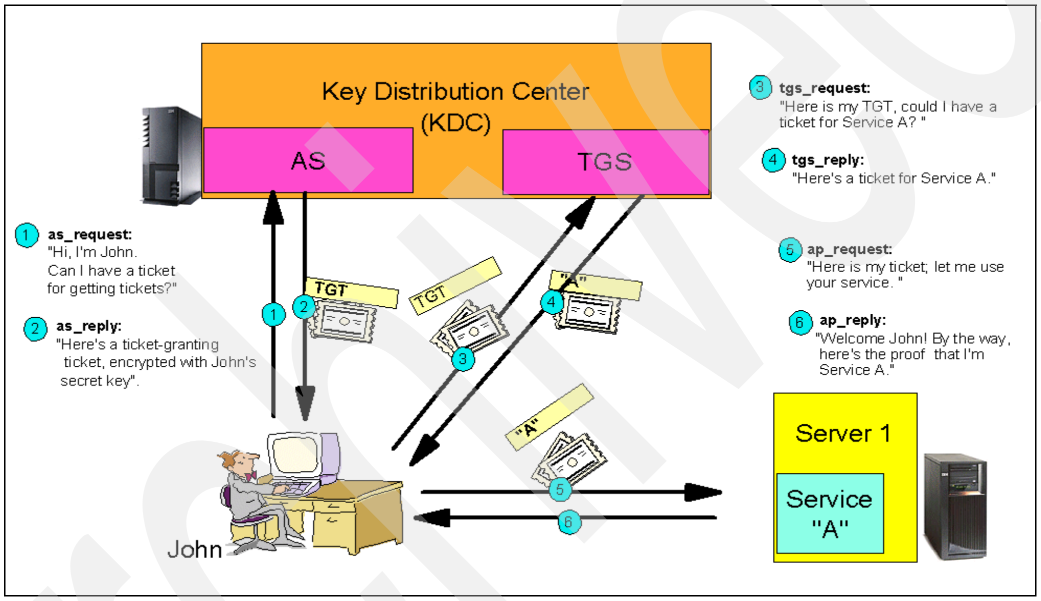


Figure 1-5 Kerberos overview

For Kerberos to be of benefit we therefore require Kerberos-enabled applications on both the desktop and the server. Not all applications are yet Kerberos-enabled. We also require the same user ID to be used for all participating applications.

1.4 SSO with Enterprise Identity Mapping

Put simply, Enterprise Identity Mapping (EIM) is a look-up table where each user's various identities in user registries (different platforms and applications) are mapped to a single identifier. EIM-enabled applications can use this table to associate the identity certified by Kerberos with the identity in its own user registry and allow the user to proceed without further challenge.

As an example, a Windows user may have been authenticated by Kerberos as `JSmith`. When that same user wants to sign on to an iSeries server where his user profile is `SMITHJ` the iSeries refers to EIM where it discovers that the source identity JSmith maps to the identifier John Smith. When EIM examines the John Smith identifier further it finds that John Smith has a target identity on the iSeries called SMITHJ. Because of this mapping we can be reasonably sure that both source (which has been authenticated by Kerberos) and the target identities are in fact the same person and that the user can be allowed the authorities of SMITHJ without further challenge.

At the latest release all IBM @server platforms can support EIM. Linux and Windows are also able to support EIM, these operating systems are enabled through a free download. The different implementations can either be downloaded in a native language or in a Java implementation and are available from the following link.

http://publib.boulder.ibm.com/eserver

Select the appropriate **Geography -> Language -> Enterprise Identity Mapping -> Server-specific EIM information -> EIM Availability and Series details**

Many IBM @server applications have been enabled to take advantage of EIM. This enablement varies by server platform and software release.

1.4.1 Why Kerberos alone is not enough

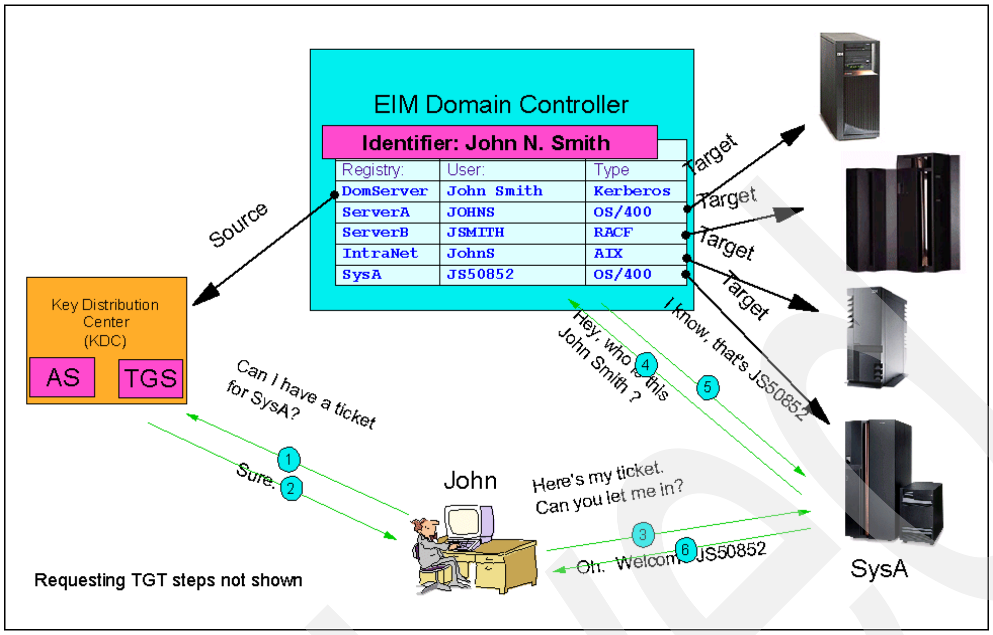
Kerberos in SSO addresses the issue of authentication for users. Administrators still have to address the issues of authorization. By itself Kerberos does not solve the problem of multiple user registries for all classes of users. It still requires that you synchronize all user IDs. This is not always possible or secure.

1.4.2 The IBM single signon strategy

IBM's single signon strategy consists of three components:

- Kerberos
- Enterprise Identity Mapping
- Enabled applications

The interaction between components is illustrated in Figure 1-6. Kerberos is used for the user authentication at initial signon. Strictly speaking, this strategy can use either Kerberos or another authentication method but those IBM applications and services which have been enabled for SSO to date have in fact been enabled for Kerberos as their authentication mechanism. Kerberos was chosen because of the large number of customers with MS Windows 2000, XP or 2003 domains which store information about users in MS Active Directory and which itself uses Kerberos. At this release the strategy therefore requires a WIndows 2000 or later desktop for the client enablement of Kerberos. It does not require WIndows to be the Kerberos server.


EIM Domain Controller
Identifier: John N. Smith
Source
Target
Target
Target
Target
Key Distribution Center (KDC)
AS
TGS
Can I have a ticket for SysA?
1
Sure.
2
John
Here's my ticket. Can you let me in?
3
Hey, who is this John Smith ?
4
I know, that's JS50852
5
Oh. Welcome JS50852
6
SysA
Requesting TGT steps not shown

Registry:	User:	Type
DomServer	John Smith	Kerberos
ServerA	JOHNS	OS/400
ServerB	JSMITH	RACF
IntraNet	JohnS	AIX
SysA	JS50852	OS/400

Figure 1-6 EIM and Kerberos together

Enterprise Identity Mapping is used to associate the initial signon with the other signons that relate to the same user and hence address the authorization issues are resolved.

Some basic client services such as Windows Network Neighborhood are already Kerberos-enabled. Server applications must be both Kerberos-enabled and EIM-enabled to use this combination and hence to participate in SSO. Several key IBM server applications and services already implement Kerberos and EIM and more are in plan. Third party developers can enable their applications via the published APIs.

IBM SSO additional tactical solutions

IBM recognizes that not all applications are as yet enabled to take advantage of the IBM SSO strategy. For the major IBM software products there are less strategic interim solutions that can achieve additional early benefits. These include solutions for IBM @server platforms at earlier releases, for Lotus Notes® and Domino and for WebSphere.

1.4.3 Possible costs of SSO with EIM

Many organizations will be aware of the benefits of achieving single signon but concerned to learn the possible costs. It is not the purpose of this redbook to analyze the costs but a few possible items are presented here for consideration.

EIM itself is included at no extra charge in major IBM @server operating systems such as OS/400, AIX® and z/OS®. It is downloadable free of charge for the Windows and Linux operating systems. EIM is implemented in IBM Directory Server which is an LDAP-enabled directory available free of charge on all IBM @server platforms and which is integrated into the OS/400 and zOS operating systems.

Kerberos is a free download from the MIT Web site and is included in all Windows 2000 and later servers. If a customer has implemented a domain on one of these servers using MS Active Directory then they already have implemented Kerberos.

There may be costs for tools to collect the user data, to collate it to enable matching and to populate EIM tables. This may highlight the need for optional user management tools but this is beyond the scope of an SSO project.

1.4.4 Benefits of EIM

There are several major benefits to an organization which chooses to use EIM in an SSO strategy.

No change to authorization models

One of the main benefits of the EIM approach to SSO is that the existing authorization models are left completely unchanged. Servers and applications can continue to manage authorities for users accessing via EIM in exactly the same way as they did before then user was added to EIM and in exactly the same way as users not yet benefiting from EIM.

No need for the same IDs

In SSO implementations that don't use EIM, the identity of all the user's IDs in every application and on every platform must be synchronized. Unless all systems run on a common application platform this can be almost impossible due to the differences in naming conventions. Even in cases where it may be theoretically possible the effort required to rename all IDs can be prohibitive.

No password synchronization (or even passwords) required

EIM does not require passwords to be synchronized between services. In fact there is no need for a password at all except at the 'entry' point of the first signon of the day. Some servers such as iSeries can be set to operate with user identities where no password is specified. This results in a significant reduction in the administrative workload.

Phased implementation

Because EIM is simply a look-up table, it does not need to be implemented in a wholesale fashion. If a user currently signs on to five servers but only three of his user identities have been mapped in EIM then those three servers can offer transparent access while the remaining two will require a user ID and password. As the IDs on those remaining servers are added to EIM then they will simply cease to challenge the user for authentication. This means that a small or pilot group of identities can be implemented in EIM based on groups of users or by server and can then later be extended to a more comprehensive solution. It is not necessary for all users, servers and applications to participate from the start.

Basis for user management

EIM provides a central collection point for all user identities. This means that, for a given user, all of his or her IDs can be tracked in one place. EIM does not itself offer user management facilities but, via the use of Application Programming Interfaces (APIs) it can be used as the basis for a user management solution, either home-grown, from IBM or from a third party software vendor.

Basis for server consolidation

Although EIM is not often considered to be a tool for server consolidation it in fact offers several advantages in this area. First, any server consolidation project needs to gather a good understanding of the users involved and the applications which they access. Otherwise, there can be no smooth transition from one server to another. Secondly, when applications move from one platform to another then new IDs and passwords will have to be set up in place of the existing ones. Using EIM, these new IDs can be associated with the actual person in advance of the switchover and the transition for the user can be relatively painless or even transparent.

1.4.5 SSO in the on demand world

In the emerging world of e-business on demand™ the changing data and application needs of the user and the business are to be satisfied seamlessly regardless of the number and location of servers delivering them.

SSO rapidly becomes a key requirement for e-business on demand infrastructure. If implemented intelligently it can enable new applications and servers to be brought into play without the user being aware of the change.

1.5 Currently enabled iSeries applications

These applications are currently capable of taking advantage of Kerberos authentication on the iSeries:

- Structured Query Language (SQL)/Distributed Relational Database Architecture™ (DRDA®)
- Distributed Data Management (DDM)
- iSeries Access for Windows and OS/400 Host Servers
- iSeries NetServer
- QFileSvr.400
- HTTP (Powered by Apache)(Available at r520 with additional PTFs)
- LDAP
- PC5250 and Telnet

These applications are also enabled for single signon using Kerberos and EIM:

- iSeries Navigator
- DRDA
- PC5250 and Telnet
- NetServer
- QFileSvr.400

2

Planning for Network Authentication Service and Enterprise Identity Mapping implementation

This chapter provides an overview of the necessary planning and preparation tasks needed to implement Network Authentication Service and Enterprise Identity Mapping (EIM) on your iSeries. Setting up Kerberos or another authentication scheme on your network is beyond the scope of this book but, since single signon (SSO) is typically a multi platform topic, we touch upon other operating systems and services as much as needed to implement our complete SSO solution.

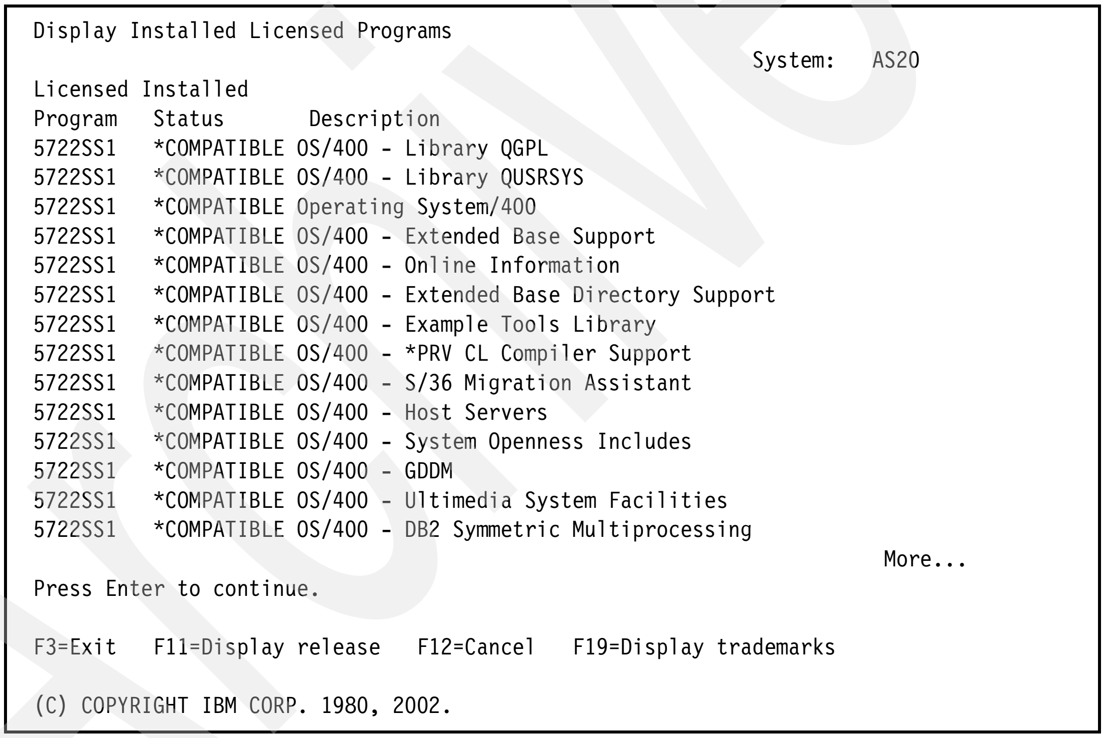
2.1 Required OS/400 components

The minimum OS/400 level you need to successfully install and configure EIM is V5R2M0. Network Authentication Service is available at an earlier release, but using the wizards provided by IBM V5R2M0 is required.

These OS/400 options and license programs are required:

- 5722SS1 Option 12 OS/400 - Host Servers
- 5722SS1 Option 30 OS/400 - QShell Interpreter
- 5722AC3 Crypto Access Provider 128-bit for AS/400
- 5722CE3 Client Encryption 128-bit (optional for Client side encryption)
- 5722DG1 IBM HTTP Server for iSeries (optional for additional scenarios)
- 5722XE1 iSeries Access for Windows

To verify that these programs are installed on your iSeries, type **go licpgm** on a command line, then select option **10**. This will get you to the display shown in Figure 2-1.



```
Display Installed Licensed Programs
                                                            System:   AS20
Licensed Installed
Program   Status       Description
5722SS1   *COMPATIBLE OS/400 - Library QGPL
5722SS1   *COMPATIBLE OS/400 - Library QUSRSYS
5722SS1   *COMPATIBLE Operating System/400
5722SS1   *COMPATIBLE OS/400 - Extended Base Support
5722SS1   *COMPATIBLE OS/400 - Online Information
5722SS1   *COMPATIBLE OS/400 - Extended Base Directory Support
5722SS1   *COMPATIBLE OS/400 - Example Tools Library
5722SS1   *COMPATIBLE OS/400 - *PRV CL Compiler Support
5722SS1   *COMPATIBLE OS/400 - S/36 Migration Assistant
5722SS1   *COMPATIBLE OS/400 - Host Servers
5722SS1   *COMPATIBLE OS/400 - System Openness Includes
5722SS1   *COMPATIBLE OS/400 - GDDM
5722SS1   *COMPATIBLE OS/400 - Ultimedia System Facilities
5722SS1   *COMPATIBLE OS/400 - DB2 Symmetric Multiprocessing
                                                                       More...
Press Enter to continue.

F3=Exit   F11=Display release   F12=Cancel   F19=Display trademarks

(C) COPYRIGHT IBM CORP. 1980, 2002.
```

Figure 2-1 Display Installed Licensed Programs

Make sure that you have the latest cumulative Program Temporary Fix (PTF) package, Hyper and Group PTFs (Database, Java, HTTP Server) installed on your system. To check the levels of your Group PTFs enter the command **wrkptfgrp** and crosscheck the information you retrieve there with the fix level information available at:

http://www-912.ibm.com/s_dir/sline003.NSF/GroupPTFs?OpenView&view=GroupPTFs

If necessary, order, install and apply any Group PTFs that are missing or not up to date. Check whether the following PTFs are applied on your System (V5R2 only):

– SI09636 OSP-INCORROUT Error calling EIM API using iSeries Navigator

– MF31494 LIC IMPROVE KERBEROS MIGRATION OPTIONS FOR ISERIES NETSERVER

The OS/400 user profile that you use for the installation needs to have *SECADM, *ALLOBJ, and *IOSYSCFG special authorities.

On your PC administration and setup client, you need to have these products installed:

– Windows 2000/XP or any other Kerberos enabled Client
– iSeries Access (Version 5 Release 2 or higher)
– iSeries Navigator (Operations Navigator in previous releases) including the Network and Security components (for administration)

It is very important to have the same iSeries Access PTF level on the client as on the server. One easy way to ensure this is by installing iSeries Access from your iSeries Server. You will find the installation files in the /QIBM/ProdData/Access/Windows/Install/Image directory in your IFS.

2.2 Required network components

To implement SSO in the same manner we have, you need a few network services, some of which can reside on your iSeries. Others, like the Kerberos Key Distribution Center (KDC) functionality need to implemented on other platforms, for example on z/OS, AIX, Linux or Microsoft® Windows 2000. For our scenarios later on in the book we assume that you have implemented the KDC on a Windows 2000 Server using the Microsoft Active Directory.

The part of Kerberos that is enabled on the iSeries is done through Network Authentication Service. Network Authentication Service allows the iSeries to use Kerberos Tickets for Authentication. However, the KDC functionality of Kerberos is not supported on the iSeries server. You need to implement this functionality on another server. The most common implementation of a KDC is Microsoft Active Directory. If you already have a Windows 2000 or 2003 Server on your network and this Windows 2000 Server is Active Directory enabled, you should use this server as a KDC. If you live in a Windows Server free environment, there are a some open source implementations of Kerberos available on Linux. We suggest that you run one of these in a Linux Partition on your iSeries server.

Checking prerequisites on your Windows 2000 Server

To have Windows 2000 server run Kerberos, a Microsoft Active Directory needs to be set up on your Windows server. We assume that you are using default Window 2000 server set-up, using Active Directory. If this is not the case, you can find information how to set up the Microsoft's Active Directory at its home page on the Internet:

`http://www.microsoft.com/windows2000/technologies/directory/AD/default.asp`

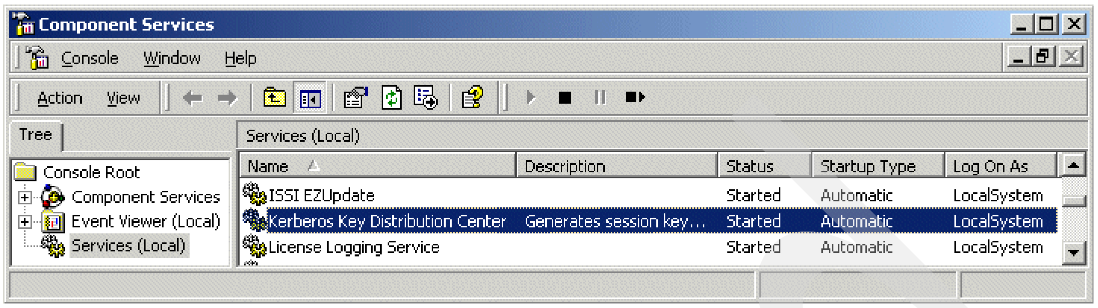
Important: If you are unfamiliar with Active Directory do not attempt to set it up in a production environment without first reading about the implications.

If you are not sure, whether your Windows domain has Active Directory set up and running, you should consult your Windows 2000 administrator. Here are two simple checks you can perform that may verify that Active Directory is configured.

1. Check if the Kerberos Key Distribution Center (KDC) service is running:

   At the console of your Windows 2000 server, select **Programs -> Administrative Tools -> Component Services**.

In the Components Services window shown in Figure 2-2, the Kerberos KDC service is highlighted. Check to see if the status is `Started`.



Name	Description	Status	Startup Type	Log On As
ISSI EZUpdate		Started	Automatic	LocalSystem
Kerberos Key Distribution Center	Generates session key...	Started	Automatic	LocalSystem
License Logging Service		Started	Automatic	LocalSystem

Figure 2-2 Check the Kerberos KDC service

2. Now check to see if there are users that have been entered in the Active Directory
   – Select **Programs -> Administrative Tools -> Active Directory Users and Computers** then highlight **Users**.

If the Kerberos KDC is not started or the Active Directory is not populated by your users you are not using Active Directory. Go to the Microsoft's Active Directory home page, listed previously, for more information on configuration.

There are additional Kerberos support tools that are not installed by default on the Windows 2000 server. We make use of the ktpass command which is included in those tools. If you do not have the tools installed on your server the process to do so is described in Appendix C, “Windows 2000 Kerberos tools” on page 235.

LDAP

Enterprise Identity Mapping requires that a Lightweight Directory Access Protocol (LDAP) server is configured with at least a basic configuration. OS/400 is an excellent platform for a directory server. Since the release of V4R3, each server has an optional LDAP-enabled directory in the form of IBM SecureWay® Directory that is known as OS/400 Directory Services. As of the release of V5R2, this has been integrated into the base of OS/400.

If an LDAP server has not been previously configured, the EIM wizard creates a basic configuration on your iSeries for you. From an EIM management point of view, you do not need to access that directory directly. This automatically implemented LDAP server is sufficient for an EIM and SSO implementation. However, should you plan to use the directory for additional functions, such as storing employee information, or configuring advanced functions, such as replication or SSL, you should first become familiar with the LDAP directory server. See "Plan your LDAP directory server" in the iSeries Information Center for planning information before you attempt to configure LDAP:

`http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm`

Select the appropriate **Geography -> Language ->Security and Directory Services -> Directory Services (LDAP) -> Getting started with Directory Services -> Plan your LDAP directory server**

If you are familiar with Directory Services and are past the planning stage for LDAP, see *"Install and configure Directory Services"* at:

`http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm`

Select the appropriate **Geography -> Language ->Security and Directory Services -> Directory Services (LDAP) -> Getting started with Directory Services -> Install and configure Directory Services**

Another excellent resource for iSeries Directory Services implementation and use is the redbook *Implementation and Practical Use of LDAP on the IBM* @server *iSeries Server*, SG24-6193.

The directory server is the container for the EIM domain and domain controller information, authorities, as well as access control to the information contained in EIM. For a production environment, we recommend that you configure the Directory Server to use SSL.

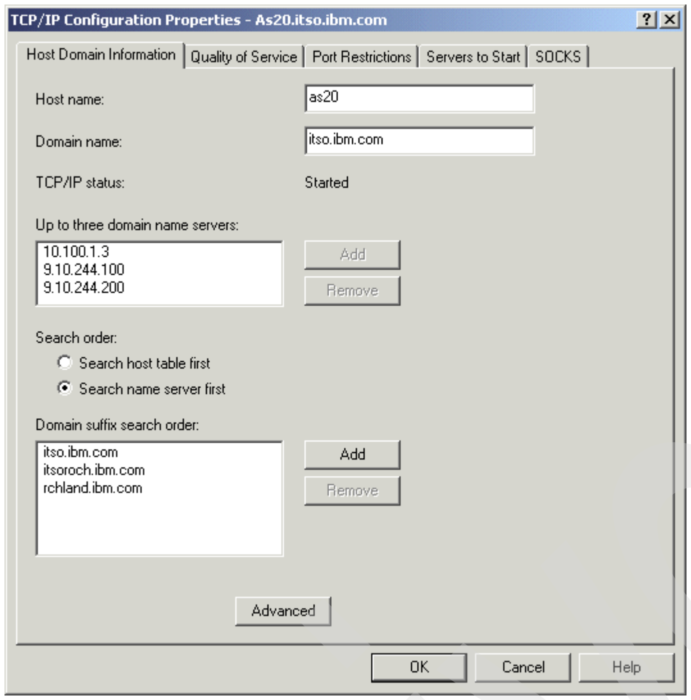
Important: Never attempt to edit the EIM files manually. Editing this information manually will result in a corrupted directory tree and you will be required to reconfigure EIM.

2.2.1 General TCP/IP considerations

The most common problem we have encountered with configuring Kerberos on the iSeries is name resolution. For Network Authentication Service and Kerberos to properly work together on your network you must ensure a reliable IP name resolution process. One of the reasons this becomes an issue so frequently is that Kerberos is case sensitive and DNS is not. With that said, the simplest method still to avoid many of the pitfalls while configuring SSO is to implement or use a Domain Name System (DNS) server on your network. A DNS server can be implemented on a variety of platforms, for our scenarios we have implemented a Bind 8.2.5 DNS server on an iSeries. For detailed information on implementing a DNS on iSeries, see chapter 8 in the redbook *iSeries IP Networks: Dynamic!,* SG24-6718.

Important: Without accurate name resolution on your network, Kerberos and EIM will fail. The error message “CWBSY1017 - rc=608 Kerberos credentials not valid on server.” means that host names are not being resolved correctly (server side). The error message “CWBSY1012 - Kerberos principal not found on server.” usually means that you probably have a name resolution problem on your client. More information on troubleshooting can be found in Appendix B, “Troubleshooting” on page 229.

To determine how your iSeries host name should be resolved, you need to check your iSeries Host domain properties as seen in Figure 2-3. To navigate there, open iSeries Navigator and select **System -> Network -> TCP/IP Configuration -> Properties.**


TCP/IP Configuration Properties - As20.itso.ibm.com

Host Domain Information | Quality of Service | Port Restrictions | Servers to Start | SOCKS

Host name: as20

Domain name: itso.ibm.com

TCP/IP status: Started

Up to three domain name servers:
10.100.1.3
9.10.244.100
9.10.244.200

Search order:
( ) Search host table first
(•) Search name server first

Domain suffix search order:
itso.ibm.com
itsoroch.ibm.com
rchland.ibm.com

Figure 2-3 iSeries host domain information

Note the host and domain name shown in Figure 2-3.

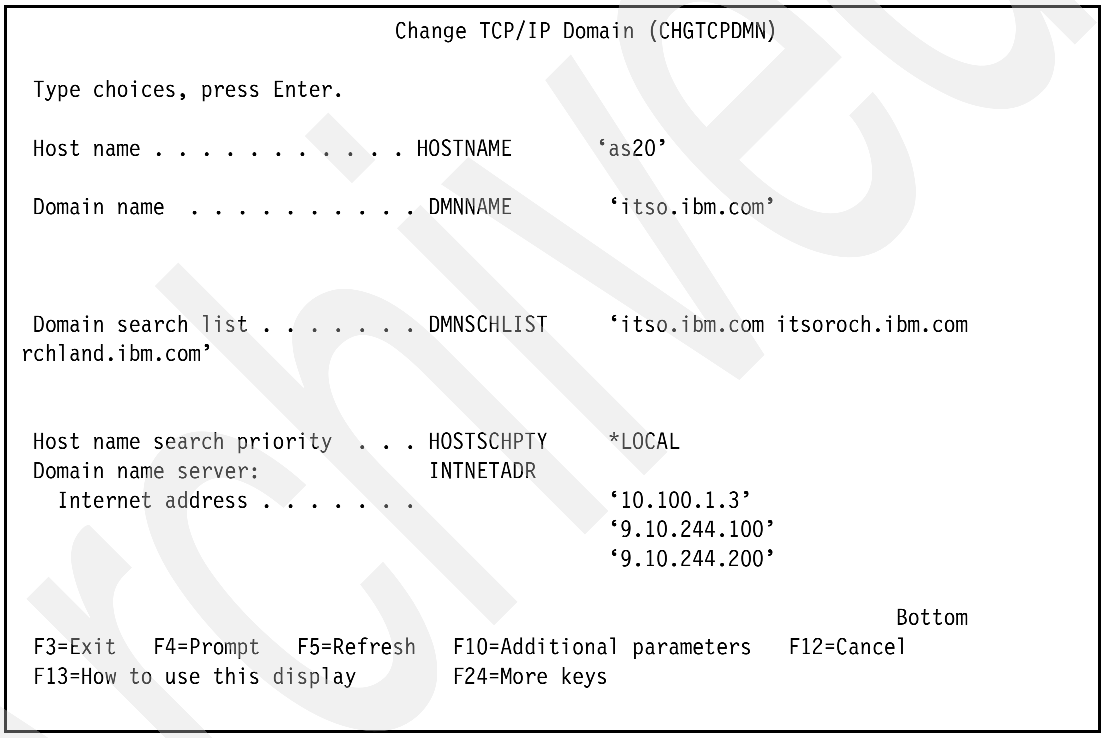
The next step we need to take is to determine how the iSeries' name is being resolved on the network. There are two possible scenarios at this point:

1. The name will be resolved on the PC in the *hosts* file. In Windows 2000 that file is located at C:\WINNT\system32\drivers\etc\hosts. On an XP system the file can be found at C:\WINDOWS\system32\drivers\etc\hosts. Open the appropriate file for your operating system and verify whether the iSeries name is listed. If so make note of the name (including upper and lower case characters) and the IP address associated with it. If the file does not exist, or there is no entry for the iSeries go to the next scenario.
2. The iSeries name is being resolved by a DNS server on the network. To determine how the DNS server is resolving the host name we have to do an NSLOOKUP (Name Server Lookup). Open a DOS window by clicking **Start -> Run...** and then typing *command* when prompted. When the command window opens type *NSLOOKUP* followed by the IP address of the iSeries. Note the name that is returned, including the case of each of the characters (upper or lower). To verify that the DNS record is complete run the NSLOOKUP command again, this time substituting the name that was returned (in the last lookup) for the IP address. If the address returned is different you will need to contact the administrator of the DNS to correct the record.

Tip: Either scenario will work but the DNS server will provide a more reliable, easy to manage, solution to your name resolution. Using DNS gives you a single point to manage your name resolution rather than having to make entries on each of the hosts in your network.

If the host name that has been noted is an exact match with what you saw in Figure 2-3, your resolution will satisfy the Network Authentication Service and Kerberos requirements. If not, change either your iSeries host name and domain name to the value returned by the NSLOOKUP function, change the value found in the *hosts* file, or change your DNS server to resolve to the name listed in the Host Domain Information window. To make the change on the iSeries, open a 5250 session to your iSeries and on the command line enter the command to Configure TCP (i**`cfgtcp)`**, then select option **`12,`** followed by **`PF11`**. By using the single quote (‘) character around your entries the case will be saved as seen in Figure 2-4. You need *IOSYSCFG and *ALLOBJ special authorities to do this.

After you have successfully configured Network Authentication Service (Chapter 7, “Enabling Network Authentication Service and Enterprise Identity Mapping” on page 97) on the iSeries you will perform a test to verify that the configuration is correct. In the test you will try to get a ticket from the KDC using the string (the following example is the string we would use in our environment) kinit `-k krbsvr400/AS20.itso.ibm.com@ITSO.IBM.COM`. Because of the way we entered the string the DNS server will return the host name *AS20* (upper case *AS*). If your host and domain name is configured similar to Figure 2-4, with the fully qualified name as20.itso.ibm.com, the kinit will fail because the iSeries is configured with lower case as in the host name.



```
                              Change TCP/IP Domain (CHGTCPDMN)

 Type choices, press Enter.

 Host name . . . . . . . . . . . HOSTNAME       ‘as20’

 Domain name  . . . . . . . . . . DMNNAME        ‘itso.ibm.com’



 Domain search list . . . . . . . DMNSCHLIST     ‘itso.ibm.com itsoroch.ibm.com
rchland.ibm.com’


 Host name search priority  . . . HOSTSCHPTY     *LOCAL
 Domain name server:              INTNETADR
   Internet address . . . . . . .                ‘10.100.1.3’
                                                 ‘9.10.244.100’
                                                 ‘9.10.244.200’

                                                                        Bottom
 F3=Exit   F4=Prompt   F5=Refresh   F10=Additional parameters   F12=Cancel
 F13=How to use this display        F24=More keys
```

Figure 2-4 Configure the host and domain name to match the DNS resolution

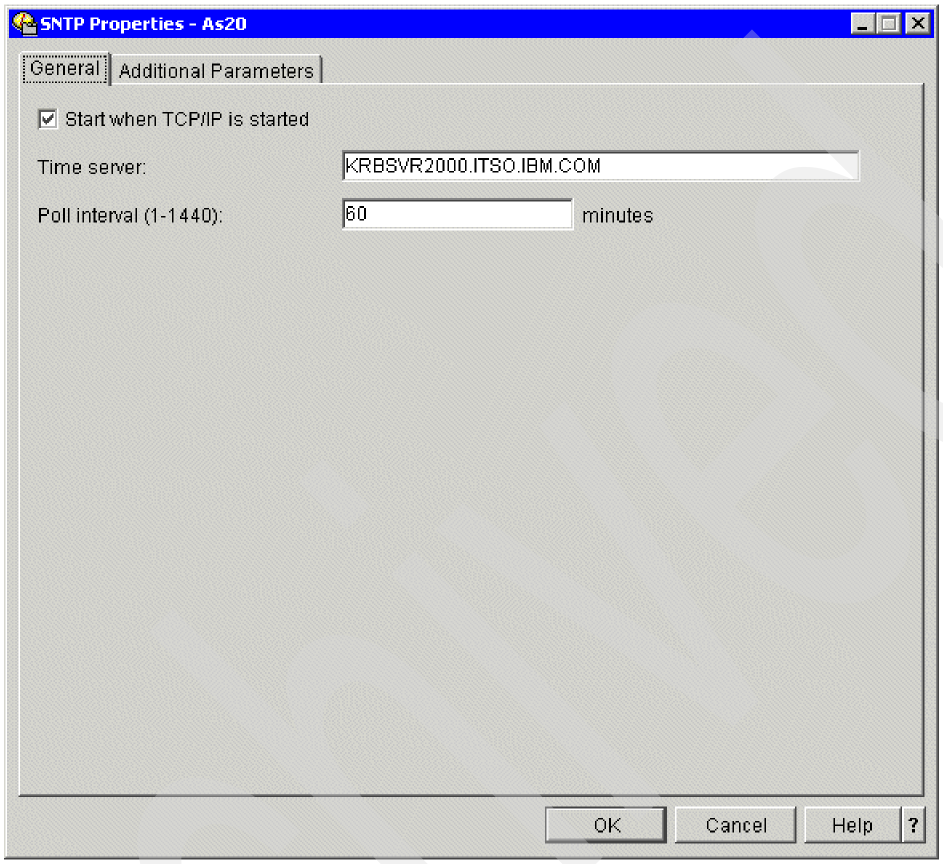
2.2.2 Time / SNTP

When you use Kerberos for Authentication in your network, setting all your servers to the correct time and time zone is a must. The maximum time skew allowed by default in Kerberos is 300 seconds. If your server time is outside this maximum skew, Kerberos authentication will not authenticate. This is a value that can be configured and can be raised to a maximum of 900 seconds in the iSeries Network Authentication Service configuration.

There are two system values that need to be checked and synchronized manually with the rest of your network:

- QTIME
- QUTCOFFSET

An easier way for keeping your system clock updated is the Simple Network Time Protocol (SNTP). SNTP is included in OS/400 5722TC1 as a client service. To configure and activate the service, select your iSeries system in the iSeries Navigator, then click **Network --> Servers --> TCP/IP** and select **SNTP** (Figure 2-5).



☑ Start when TCP/IP is started

Time server: KRBSVR2000.ITSO.IBM.COM

Poll interval (1-1440): 60 minutes

Figure 2-5 SNTP configuration Step 1

Enter the fully qualified name of a SNTP server into the time server field above. This could either be an external (Internet) SNTP server, your Active Directory controller or any other SNTP server. We used the same system that the KDC resides on. The default polling Interval is 60 minutes, this should do for most implementations. Click OK to continue.

Your iSeries will now poll the SNTP server every 60 minutes and adjust its software clock.

2.3 Planning your EIM implementation

When planning Enterprise Identity Mapping there are several areas of concern that need to be addressed before you start. One of the most critical choices you must make is the system you will use to be your domain controller. Along those same lines will be questions about how to organize the domain itself, the organization of administrators, identifiers, and associations. We address the basic concerns in the following section and provide more detailed explanations in Chapter 6, “Enterprise Identity Mapping” on page 71.

2.3.1 Selecting the system to act as the domain controller

All of the IBM @server platforms, when running at the most current release with up-to-date fixes, are capable of being your domain controller. Furthermore, all of the platforms support the EIM client APIs. However only two of the operating systems currently have EIM administration tools, those being OS/400 on the iSeries and z/OS on the zSeries®. For the purposes of this book, we focus on all facets of EIM running on the iSeries.

Defining the domain

One of the first concerns to address is how you will define your domain. Each domain can represent the entire enterprise, a department, or a single application. To get the most out of your effort, you should consider grouping as many applications together as possible. The benefit of that approach is that you will have a single central repository to manage.

Performance considerations

Depending on the number of lookup operations being done you also need to consider the performance aspect. If you expect to have a large number of lookups you may want to replicate the controller to secondary systems to handle some of the load. If you have one system that does more lookups than others you may give some thought to moving the primary domain controller to that system. The resulting benefit of that move would be improving lookup response time and reducing network traffic.

Accessing the domain controller

Authorities required to access and administer the domain controller are discussed briefly in the following section, 2.3.2, “Administering EIM”, and in-depth in 6.3.8, “Setting Up EIM Authorities” on page 87. As always, network security is a factor in your decision making. When making the initial connection (binding) to the EIM controller the client provides a distinguished name (DN) and password to use for authentication. By default we will do a simple bind, which is the weakest form of authentication, in which the DN and password are sent in plain text over the network. There are several methods available to secure this transaction including challenge-response authentication mechanism (CDRAM-MD5), server authentication with Secure Sockets Layer (SSL), client authentication with SSL, or Kerberos authentication. The level of authentication you use will be dependent on the requirements of the application and your own network security.

2.3.2 Administering EIM

Depending on the complexity of your environment EIM can be tailored to be administered by a single person or a group of individuals with specific skills. Because of the possible interaction with so many other software products it may not be possible for a single individual to configure and administer EIM. There are several predefined roles in EIM that can be assumed by a single individual or assigned to individuals in a team. Below is a list of the roles that are provided in the administrative Graphical User Interface (GUI) of iSeries Navigator:

- EIM administrator
- Identifier administrator
- EIM mapping operations
- Registry administrator
- Administrator for selected registries

If your enterprise currently has an LDAP administrator, system administrators or programmers, they will have to be involved in the implementation and maintenance of EIM also.

2.3.3 Naming conventions

There are two naming schemes that need to be addressed in EIM. The first is the EIM identifiers and the second is the EIM registry definitions. Remember that EIM does not need to be implemented across the entire enterprise simultaneously, but you should plan your naming conventions prior to implementation to provide consistency throughout your organization.

EIM identifier naming

The size of your organization may determine the naming scheme that you use for the identifiers. Each identifier is associated with a user and must be unique. If you have a small company you may be able to use the individual's name. Larger organizations will have to devise a naming convention to cope with identical names across the enterprise. The iSeries Navigator GUI does provides a method to automatically produce unique identifiers. If that option is selected and you try to create an identifier that already exists the duplicate identifier name is created with an eight byte timestamp appended to the name before adding the identifier to the domain.

EIM registry definition naming

EIM maps users from one registry to an equivalent user in another registry. There needs to be a user registry created for every operating system and application that participates in the EIM domain. You can create a registry definition for user registries that exist on any platform. Any application that is EIM enabled must have a registry defined. The only limitation to the number of registries you may define is the size of your LDAP directory. The application can refer to the registry by its defined name or by an alias. Each registry can have multiple aliases. When naming your registry definitions you should make sure the name is descriptive enough that the EIM administrators can easily identify the user registry the EIM registry definition is linked to.

2.3.4 EIM associations

EIM associations represent the relationship between an EIM identifier and a user defined in a specific user registry. For example, we have an identifier named Patricia Boats. Patricia's OS/400 user ID is PATRICIAB. Patricia signs on to her Windows 2000 computer with the user ID of pboats. In the Kerberos realm pboats receives a ticket granting ticket from the Kerberos server. When Patricia wants to open an emulation session to the iSeries she requests a Kerberos service ticket. That service ticket is then passed on to the iSeries with the user ID of pboats on it. The user ID from the Kerberos registry, pboats, will be used as a source association. When EIM does a lookup it finds that the user pboats has a source association with the EIM identifier Patricia Boats. Patricia Boats has a target association of PATRICIAB on the iSeries. Since the source ID has already been authenticated the emulation session will be started with the PATRICIAB signed on with all of that user's normal authorities. Each association is created to map a specific user to a particular application. The number of associations is only limited by the size of the directory server that the EIM domain is stored on, but for ease of management you do not want to create associations that will never be used. The applications that the associations are created for should document the types of associations that are required.

2.4 Information to collect before you start

In Table D-1 on page 240 we have provided a worksheet that details all of the prerequisites you need to collect to successfully configure Kerberos and EIM. Additionally, Table D-2 on page 241 contains a blank configuration worksheet for you to complete. In Table 2-1, that follows, is an example of our completed worksheet for the basic configuration that we used in our scenario. Note the *Item* column on the left. That item value will be referenced in Chapter 7, “Enabling Network Authentication Service and Enterprise Identity Mapping” on page 97. When you are prompted to provide one of these values we provide the letter (A,B,.....S) of the item that corresponds to the input required.

Table 2-1 Our completed planning worksheet

Item	Information to collect	Result
A	What is the name of the Kerberos default realm to which the iSeries will belong?	ITSO.IBM.COM
B	What is the KDC for this Kerberos default realm?	KrbSvr2000
C	What is your KDC’s fully qualified host name?	KrbSvr2000.itso.ibm.com
D	What is the port on which the KDC listens?	88
E	What is name of the password server for this KDC?	KrbSvr2000
F	What is the port of your password server?	464
G	What is the password for your iSeries service principal(s)?	win4IBM
The following items will be used to create the iSeries principal on the KDC		
H	What is the name of the Kerberos principal?	krbsvr400 (when creating the iSeries principal this name must be used)
I	What is your iSeries host name?	as20
J	What is the fully qualified host name of the iSeries?	as20.itso.ibm.com
K	What is the name of the Kerberos default realm to which the iSeries server belongs? (Default: domain name converted to upper case)	ITSO.IBM.COM
L	What is the full name of the Kerberos principal? (`krbsvr400/fully.qualified.host.name@YOUR.KERBEROS.REALM`)	krbsvr400/as20.itso.ibm.com @ITSO.IBM.COM
M	What is the password / shared secret for this principal? (Must be the same as item G)	win4IBM
The following items will be used to configure Enterprise Identity Mapping (EIM)		
N	Which type of basic EIM configuration do you want to create on your iSeries system? ▸ Join an existing domain ▸ Create and join new domain	Create and join new domain
O	Where do you want to configure your EIM domain, or what EIM domain you want to join?	as20.itso.ibm.com
P	What is the name of the EIM domain you want to create or join?	ITSO EIM
Q	Do you want to specify a parent DN for the EIM domain? If yes - specify the parent DN	NO

Item	Information to collect	Result
R	What is the administrator distinguished name (DN) on the LDAP server which will be used as the EIM domain controller?	CN=administrator (see the setup instruction R for notes.
S	What is the administrator password on the LDAP server will be used as the EIM domain controller?	ldappw

Filling in the Worksheet

Here are the directions about how to find each of the parameters to complete the worksheet.

- **A** - Under normal conventions the Kerberos realm will be the domain name in upper case. This is not always the case though, to verify on the KDC, right-click **My Computer** and select **Properties**. Click the **Network Identification** tab. Whatever the domain name listed in this window is, converted to all uppercase, is the Kerberos realm name.
- **B** - This is the host name of the KDC for the default realm. On a Windows 2000 server desktop, right-click **My Computer** and select **Properties**. Click the **Network Identification** tab. In this window you see the Full computer name which includes the domain name. The host name is everything prior to the first period (.). For example our Full computer name is as20.itso.ibm.com, the host name is as20.
- **C** - The KDC's fully qualified host name can be found in the same manner as B. *The Full computer name* listed on the Network Identification tab is the value item C represents.
- **D** - By default Version 5 Kerberos listens on port 88. This may be changed so it is best to verify before proceeding.
- **E** - This will default to the same value as item E. If your password server is different than the KDC you can specify the host name in place of the default value.
- **F** - The default port is 464, unless you have changed the configuration of the password server this should be correct.
- **G** - This is the shared secret between your iSeries principal and the KDC, this value must match item O.
- **H** - Regardless of your environment this value must be krbsvr400.
- **I** - In iSeries Navigator select your **System -> Network** and then right-click **TCP/IP Configuration** and select **Properties.** When the Properties window opens it is on the Host Domain Information tab, the first parameter in that window is *Host name*, that is the name we want to use for this configuration.
- **J** - Also found in the Host Domain Information window is the *Domain name* field which immediately follows the Host name. Item J is a combination of the host name and the domain name. The fully qualified name of your iSeries is simply the host name, followed by a period (.) and then the domain name.

Tip: Something to watch for (we have seen this on many occasions) in option 12 of the Configure TCP/IP menu, is the host name that has already been entered in the domain name field. For example:

```
Host name: as20
Domain name: as20.itso.ibm.com
```

Following that logic would give us a fully qualified name of `as20.as20.itso.ibm.com` which probably will not resolve at all.

- **K** - Normally this will be the domain name of that the system resides in converted to uppercase letters. This should be the same as item A.
- **L** - This value is used to request tickets and is case sensitive. Remember that the principal name for the first scenario must be krbsvr400, followed by the domain name, followed by the Kerberos realm name.
- **M** - The shared secret must be the same as item G. This value will be used on the KDC when you are mapping the principal to the user.
- **N** - If you do not have LDAP configured already you will select *Create and join new domain*. If you do have an Directory server configured that you want to use as a domain controller you can specify *Join an existing domain*.
- **O** - If you chose to create a new domain (item N) then specify the fully qualified domain name of the iSeries (item J). If you are joining an already existing domain, specify the fully qualified domain name of that Directory server.
- **P** - This is an arbitrary value, as you can see we named ours ITSO EIM.
- **Q** - If you want to further define the location of the EIM data you may specify a parent DN by selecting Yes when you reach this point in the wizard.
- **R** - We recommend that you create a new LDAP user and add it to the EIM Admin group. Use this new user ID and password for the system. Creating LDAP users is outside the scope of this redbook, so we will use the LDAP administrator's ID and password instead. If your are currently using LDAP contact the administrator of your Directory server to obtain this value.
- **S** - The password is up to you if you are creating a new configuration. Again, if you are using an existing Directory server you will need to contact the administrator of that server to get the password.

3

The redbook example scenario

The real advantage of using single signon is being able to enable users to transparently access systems without having to log in repeatedly. A by-product of this is simplifying administrative procedures. Due to the fact that Enterprise Identity Mapping (EIM) is an authorization mechanism not an authentication mechanism we are also implementing network security in parallel. With this in mind we explain EIM planning, implementation and management of users. We also show you how to use the implemented infrastructure to enable applications to use EIM and Kerberos.

This chapter provides an overview of the scenario which has become the basis for this book. We outline the different objectives involved at each stage of the project and how these objectives are to be accomplished.

3.1 Scenario overview

In our scenario we have a company called *The Bike Shop.* The company uses a network of Windows 2000 client machines with a Windows 2000 server. In addition to the Windows 2000 server The Bike Shop also has two iSeries servers. These iSeries servers are used extensively to meet the changing needs of the business.

The first iSeries is used to hold business information such as stock management information and customer information. This information is stored in DB2® databases. On this iSeries there is an Enterprise Resource Planning software package which controls the stock management and customer information. There are also RPG applications which create reports based on the information from DB2.

The second iSeries runs Lotus Domino as a mail server internally and also has Domino Web Access installed to allow remote users access to their e-mail. As well as running Domino the iSeries also uses WebSphere Application Server to provide access to products and services offered by The Bike Shop. This is served to the browser through an Apache Web server.

The company has 100 users and each user has multiple user names and passwords. Each user has IDs on these systems:

- Windows
- Lotus Notes
- WebSphere
- Both iSeries systems

In addition to the users in the office the company also has users which work remotely. At present these remote employees connect through VPN in order to access the iSeries applications and sensitive information on the iSeries servers. When remote users are not connected to the network through a VPN connection they can quickly access their e-mail through a Web browser using iNotes. This is useful for sales representatives travelling to customer sites where a VPN cannot be obtained.

Customers can also access product information and services over the Internet.

For a graphical example of how the The Bike Shop network infrastructure looks, see Figure 3-1.

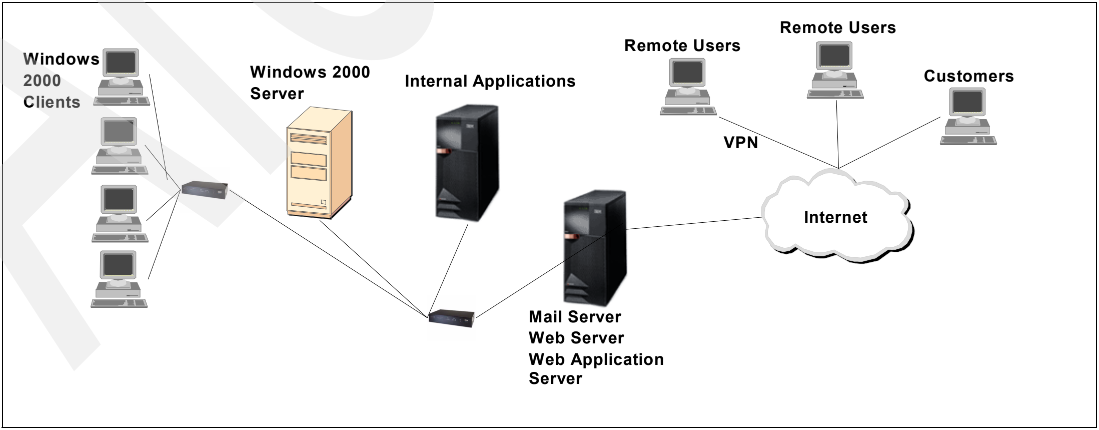


Figure 3-1 Scenario network infrastructure

3.2 Objectives

The company is growing rapidly and it realizes that its current administration practices are not coping with the needs of the business. Although they have a windows domain for users they are not making much use of the services offered by the hardware available to them.

The primary objective that the organization wants to achieve is as much single signon enablement of applications and services as possible. The organization understands that the current configuration of the company's infrastructure will have to be examined carefully and possibly redesigned in order to implement an effective single sign on solution. These are the steps we followed in order to implement an effective single signon solution.

3.2.1 Make effective use of Kerberos

Currently The Bike Shop has a Windows 2000 server which comes with active directory installed and without the company knowing it the Windows 2000 server also functions as a Kerberos Key Distribution Center. The goal is to set up this Key Distribution Center as a basis for authentication using EIM.

3.2.2 Network Authentication Service

Not only does The Bike Shop have a Windows 2000 server but they also have two iSeries servers. These iSeries servers will also need to be configured so that they can participate in the Kerberos realm set up by the Windows 2000 server. This is done by enabling the Network Authentication Service. Setup of the second iSeries is demonstrated in 8.3, "Enabling another iSeries server for single signon" on page 145.

3.2.3 EIM in action

After the EIM infrastructure has been set up and populated with information from the other servers the next objective for the company is in fact the first benefit the company will see. This objective is to allow EIM enabled applications that come with the iSeries Navigator and the iSeries Access to use the implemented EIM infrastructure. Single signon for Kerberos enabled and EIM enabled applications will now become a reality for The Bike Shop. For a list of applications which support Kerberos and EIM, see 1.5, "Currently enabled iSeries applications" on page 13. For information on how to implement these supported applications, see Chapter 8, "Other scenarios" on page 127.

3.2.4 Managing users in EIM

Now that The Bike Shop has created the infrastructure and populated EIM the next step is the continual management of this infrastructure. With current technologies implemented in organizations there are books on situations that the administrator will probably face while they are dealing with a product. With EIM there are at present no commercial books to refer to and these situations will only be able to be performed by the administrator after a solution has been identified. We have mentioned a few possible points in 6.6, "EIM User Management" on page 92.

3.2.5 Backing up EIM

With the organization moving to EIM it puts a lot of trust with the administrators to manage the systems effectively and to conform to the organizations procedures and rules for dealing with computers. Of course some things are out of the control of the administrators, such as in the event of a power failure causing hardware to fail in an iSeries server. In the event of this happening responsibility lies with the administrators to create another EIM infrastructure. This would be made infinitely easier if the information was backed up.For more information on how to backup up the information in EIM see Appendix A, “Backup and recovery” on page 225

3.2.6 Kerberos enabling an application

The EIM infrastructure has now been created and is functioning correctly. In order to further their single signon infrastructure The Bike Shop would like to Kerberos enable other existing applications to allow them to authenticate using Kerberos tickets. For more information on Kerberos enabling an application see:

http://java.sun.com/j2se/1.4.2/docs/guide/security/jgss/tutorials/

3.2.7 EIM enabling an application

The Bike Shop, in addition to client side applications, also have a multi-tier application which connects to a program running on the server. This server application runs SQL queries against information held in DB2 and returns it to the client. A problem is that when a user wants to connect to the DB2 database they are challenged for a user name and password to connect to the database. The Bike Shop would like to EIM enable this application so that a user does not have to log onto the DB2 database in order to run this query. For this example of EIM enabling an application, see Chapter 8, “Other scenarios” on page 127

In addition to The Bike Shop having this application which they want to EIM enable they also have an RPG application which they would like to convert to use EIM. The problem that they have is that there are three different prices stored in the stock database and depending on what user is accessing the database depends on which price is returned, retail, wholesale or cost price. The Bike Shop wants to use EIM so that a user ID is only needed for each price that is to be returned. This *task* user name will be associated to the users identity in the EIM domain. After logging in EIM code looks up the user name for this user on the target system and then calls the application using this user name. Only the price lists which the user is authorized to access would be returned. For external users who quickly want to access back end information from sites where the possibility to dial in via a VPN is not possible this is would be a very powerful way to access important information effectively. See “The Bike Shop scenario” on page 128 for details on this scenario.

3.2.8 A second iSeries

In this scenario for The Bike Shop there is are two iSeries. Naturally when the organization wants to implement this EIM configuration they would like all computers that they have in the network to be EIM enabled. Another object for this scenario is to enable the iSeries used for Web and e-mail functionality to participate and use information in the EIM domain. See 8.3, “Enabling another iSeries server for single signon” on page 145, and 8.5, “Enabling Domino Web Access for single signon and EIM” on page 162 respectively for our implementation.

Part 2

Building blocks for single signon and Enterprise Identity Mapping

In the following three chapters we describe the technology that is used to implement a single signon solution based around Enterprise Identity Mapping. There are two main components of the solution to single signon, these are a method of authentication and also a method of authorization. The method of authentication that we explain in this book is Kerberos. After Kerberos we explain about the Network Authentication Service and how it links in with Kerberos to use services on the iSeries. We then discuss EIM in depth and how this becomes the authorization segment in the single signon solution provided by both Kerberos and EIM.

Kerberos is not the only authentication protocol available to us however it is the most widely available and supported. It is imperative to understand that single signon is not just limited to this authentication protocol.

4

Kerberos Network Authentication

This chapter provides an overview of the Kerberos protocol focusing on how Kerberos fits into the single signon solution. We describe the components of Kerberos and explain how they all fit together. This chapter is not meant to instruct you on how to implement Kerberos but rather give you some reasons why it should be considered.

4.1 An introduction to Kerberos

The Kerberos architecture was designed and developed in the 1980s by the Massachusetts Institute of Technology (MIT), as part of the Athena project. Its name derives from Cerberus, the legendary three-headed dog of Greek mythology, which guarded the gates of the underworld. He made sure that only the souls of the dead could enter Hades and that no souls could escape. He could be said to be the original authentication mechanism.

The current version of Kerberos is Version 5, which is standardized in RFC 1510, The Kerberos Network Authentication Service (V5). For details, see

http://www.ietf.org/rfc/rfc1510.txt

Kerberos is one possible solution to your network security problems. It provides the tools of authentication and strong cryptography over the network to help you secure your information systems across your entire enterprise. Kerberos authentication itself does not automatically imply that an entire session is encrypted. However, Kerberos enables a secure exchange of encryption keys that could be used by a client program for session encryption. iSeries Access, for example, does not implement the encryption part of Kerberos but iSeries Access traffic can be encrypted by SSL instead.

These are the different questions that we answer in the overview of Kerberos:

- What is Kerberos and why is it needed?
- Why is Kerberos useful?
- Where does Kerberos fit into the single sign on solution?
- What components are involved in Kerberos?
- How do these components fit together to provide authentication and security?

It is also important to understand what Kerberos cannot do or is not defined to do in order to know what else is needed to work with Kerberos in an organization's infrastructure.

4.1.1 The need for Kerberos

Kerberos has grown out of the need for a secure protocol which allows users to identify themselves to applications and services. Normally in a network, information is transmitted across the network in an insecure form, this is usually just plain text. This presents a security problem for many organizations as passwords and confidential data are transmitted without any security on the network. Hackers, whether internal or external, have the opportunity to write malicious programs that can take advantage of these plain text transmissions on the network. By monitoring the contents of packets as they are transmitted across the network hackers are able to intercept passwords, read and change data that may be sensitive to the operation of the business. Kerberos removes many of the risks by encrypting passwords, reducing or eliminating the use of passwords, and providing a method to check the data integrity, as we will see later in the chapter.

4.1.2 Kerberos versions

The two main open source implementations that we discuss in this section are:

- MIT's Kerberos
- Heimdal

MIT Kerberos

The Massachusetts Institute of Technology (MIT) implementation of Kerberos is currently at V5 with a release version of 1.3.1. In January of 2000 and again in October of 2000 the US export laws for open source cryptography were relaxed. This means open source code and objects relating to this open source code are now freely exportable from the USA. MIT's download page however still asks questions pertaining to the laws before they were relaxed, MIT is doing this until they can consult legal council over the widespread download of their binaries. In the meantime a version of the MIT Kerberos implementation can be downloaded from the MIT distribution Web site located at the URL:

`http://web.mit.edu/kerberos/dist/index.html`

When flaws are found in the Kerberos implementation MIT produces a Security Advisory which lists a summary of the problems, the impact that this has on the protocol, where to get fixes, and what the fixes actually change in the Kerberos implementation in order to remove the weakness. The advisories can be found at:

`http://web.mit.edu/kerberos/www/advisories`

Heimdal

This version of Kerberos is also at V5 and the release of this version is 0.6. The goal of Heimdal is to produce a free version of Kerberos that can be used by anyone. It aims to be compatible with the MIT Kerberos V5 API and also the Generic Security Services API (GSS). Information about the GSS-API can be found in Chapter 9, "Programming APIs and examples" on page 187. The Heimdal version of Kerberos is a cleanroom implementation of Kerberos, meaning that it has been built from the ground up and although it has the same functionality as the MIT version of Kerberos the underlying code in each implementation is unique. Both Heimdal and MIT resolve the same bugs in their implementations within very similar time frames.

Kerberos compatibility between versions

Since the implementation of the Version 5 Kerberos protocol more weaknesses with the Version 4 protocol have been detected. Many of the differences between the two protocols are because parts of the Kerberos implementation have been re-written using more standardized protocol formats (such as ASN.1 (Abstract Syntax Notation One)) in order to make the implementation more easily extendible in the future. The only truly safe way to stop compatibility bugs appearing in Version 5 is to disable the Version 4 support for Kerberos in the Key Distribution Center (KDC). The KDC will be discussed later in this chapter. We recommend only using the Version 5 Kerberos protocol if implementing Kerberos. If you are working with Kerberos Version 4 we would suggest migrating to a more stable release of Version 5. Some new features that appear in Kerberos Version 5 are:

- The key salt algorithm has been changed to use the entire principal name.
- Kerberos tickets can now contain multiple IP addresses and addresses for different types of networking protocols.
- A generic cryptography interface module is now used, so other encryption algorithms beside DES can be used, this is a major weakness with the Kerberos 4 protocol.
- There is now support for replay caches, so authenticators are not vulnerable to replay.

4.1.3 Authentication versus authorization

One of the most important aspects to learn in single signon is that there are two mechanisms which work together to achieve this goal. The most common misconception we have found is identifying and understanding the difference between authorization, whether a user is allowed to do something, and authentication which is concerned with the identity of the user. These two mechanisms are independent of each other but come together to provide an ideal solution to this single signon problem. As you will see in the sections that follow, Kerberos is not concerned with the authorization of the user it will only present the identity of the user to the service that it wants to access. The authorization of the user is left up to the service itself to decide what operations can or cannot be performed.

It is important however to have a good combination of authentication and authorization. What would be the point of having excellent authentication in a network if there was poor authorization in the organization? For example, Kerberos authentication coupled with an initial sign-on which allowed zero length passwords, the user would only have to log on using a username to impersonate that user completely and be unchallenged.

4.2 The components of the Kerberos protocol

In this section we describe the different components that fit into the implementation of the Kerberos protocol. These are:

- Tickets
- Principals and Realms
- Key Distribution Center (KDC)
- Services

Figure 4-1 shows how all the different Kerberos components fit together to perform a Kerberos network solution.

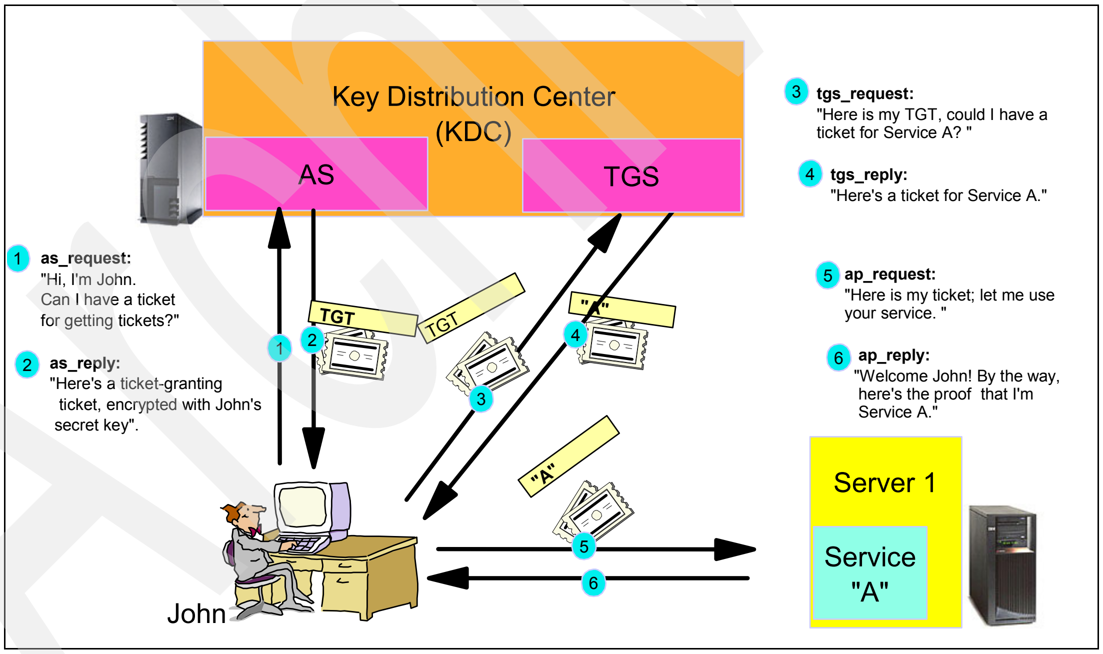

Key Distribution Center (KDC)
AS
TGS
1 as_request: "Hi, I'm John. Can I have a ticket for getting tickets?"
2 as_reply: "Here's a ticket-granting ticket, encrypted with John's secret key".
3 tgs_request: "Here is my TGT, could I have a ticket for Service A? "
4 tgs_reply: "Here's a ticket for Service A."
5 ap_request: "Here is my ticket; let me use your service. "
6 ap_reply: "Welcome John! By the way, here's the proof that I'm Service A."
TGT
TGT
"A"
"A"
1
2
3
4
5
6
Server 1
Service "A"
John

Figure 4-1 Kerberos overview

4.2.1 Kerberos Tickets

The word ticket is used to describe how authentication data is transmitted in the Kerberos environment. Tickets are essentially an encrypted data structure which use shared keys issued by the KDC to communicate in a secure fashion. Where the ticket has been created from within the KDC the defines the purpose of the ticket. When a principal requests the initial authentication the ticket comes from the Authentication Server (AS) see Figure 4-1. This ticket is called the Ticket Granting Ticket (TGT) and this ticket is used to request tickets for other services on the network. When the principal wants to use a service on the network the TGT, along with a service request is sent to the Ticket Granting Server (TGS). The TGS responds by issuing a Service Ticket (ST). The steps of creating and issuing tickets is covered in depth in 4.2.3, “The Key Distribution Center” on page 40. Each ticket can also contain options that give you more control over the use of the tickets. Options that have been added in Kerberos Version 5 are:

- Can the ticket be forwarded? For example passed onto other back end services to retrieve information.
- Is the ticket renewable? This forces a ticket with a long life time be renewed periodically from the KDC.
- In the case of a post dated ticket or an expired ticket, can the ticket be used?
- There are also proxy tickets which can be used to tell a service that the network address in this ticket is different from the address in the initial ticket granting ticket.

Note: A complete list of options (flags) can be found in Section 5.3.1 of RFC 1510 at the URL:

`http://www.ietf.org/rfc/rfc1510.txt`

4.2.2 Principals and realms

To understand how principal names are generated in Kerberos, we must first understand Kerberos realms. A Kerberos realm is often referred to as an administrative domain. A realm consists of members, which can be users, servers, services or network resources that are registered within a KDC’s database. Each of these members has a unique identifier that is called a principal. The Kerberos realm is made up of the KDC and all of its principals.

Important: The normal conventions of naming Kerberos realms is to create them using uppercase.

Again, the principal is a unique identifier that the KDC can assign tickets to. There are three parts of a principal name, the primary, the instance (optional) and the realm. The primary component of the principal name is separated from the instance by a forward slash (‘/’) The realm is separated from the rest of the principal name by the at sign (‘@’).

- The primary part of the principal name can be either the user’s name, or the word *host,* or the name of the service. An example of a principal in the ITSO.IBM.COM realm would be `cwayman@ITSO.IBM.COM`.
- The instance component is optional and normally not used for an individual principal. The instance is used to qualify the primary, for example if cwayman was the administrator of the KDC’s database he would have an additional principal that looks something like `cwayman/admin@ITSO.IBM.COM`. Note that the principal `cwayman` and `cwayman/admin` are two completely separate principals with different passwords and possibly a different set of authorities. The instance component is also used when specifying a host or a service principal. It is the fully qualified domain name of the host, for example `telnet/as20.itso.ibm.com@ITSO.IBM.COM`.
- The realm portion of the principal’s name is the name of the Kerberos realm which, as stated previously, is usually the domain name in uppercase.

4.2.3 The Key Distribution Center

In the Key Distribution Center (KDC) there are actually three different functions which are performed. There is an Authentication Server (AS), a Ticket Granting Server (TGS) and the KDC Database which holds shared secret information about each principal in the Kerberos realm. Since these functions are incorporated into the KDC, and are also linked by the services provided to the principals, they usually reside on the same machine.

The Key Distribution Center Database

When a principal is created a shared secret is created that only the KDC and the created principal know about. This shared secret is used to encrypt communications between the principal and the KDC. This shared secret is never transmitted in plain text. The KDC database contains all of the principals in the realm and their associated secret keys. There is no requirement that this database reside on the same physical machine that KDC is on but it is considered good practice to store the information on the same system.

Functions of the Key Distribution Center

The two primary functions of the KDC are:

- Granting a Ticket Granting Ticket
- Granting a Service Ticket

After reading this section you will be able to understand the full communication protocol that is followed when two principals want to communicate with each other as well as the security involved in the interactions.

Receiving a Ticket Granting Ticket

When a principal logs on to either his computer or a *kerberized* application for the first time either that day or since a previous ticket he has received has expired an interaction takes place between the client and the Authentication Server.

Kerberized: This is the term given to services which use Kerberos for authentication.

In this interaction the principal sends a request (AS_REQ) to the AS based on their identity and who they want to talk to, in this case it is a principal and the TGS. This information is transmitted in plain text, the AS then based on the identity of the principal builds a Ticket Granting Ticket (TGT), and replies to the principal encrypting the ticket using the shared secret of the principal who requested it. This response is known as the AS_REP. There is no encryption between the principal and the AS initially because the authentication server must first discover the identity of the principal to determine the shared secret. Once the identity is known to the AS it responds with a TGT that has been encrypted using the principal's shared secret. This was a downfall of Kerberos Version 4 as the ticket could be captured and then an offline brute force attack could be performed to find the principal's secret key.

Kerberos 5 introduces a method called pre-authentication which we will see later in this chapter. The request and reply between the AS is detailed in Figure 4-2. The encrypted ticket can be decrypted by the authentic principal and used to obtain services on the network such as the telnet example described in the kerberized box above. Even though the principal can decrypt the TGT given to him by the AS there is still another level of encryption beneath this level. This extra level of encryption is performed using the shared secret of the AS and the Ticket Granting Server (TGS). We will see why there is this extra encryption in the following section.

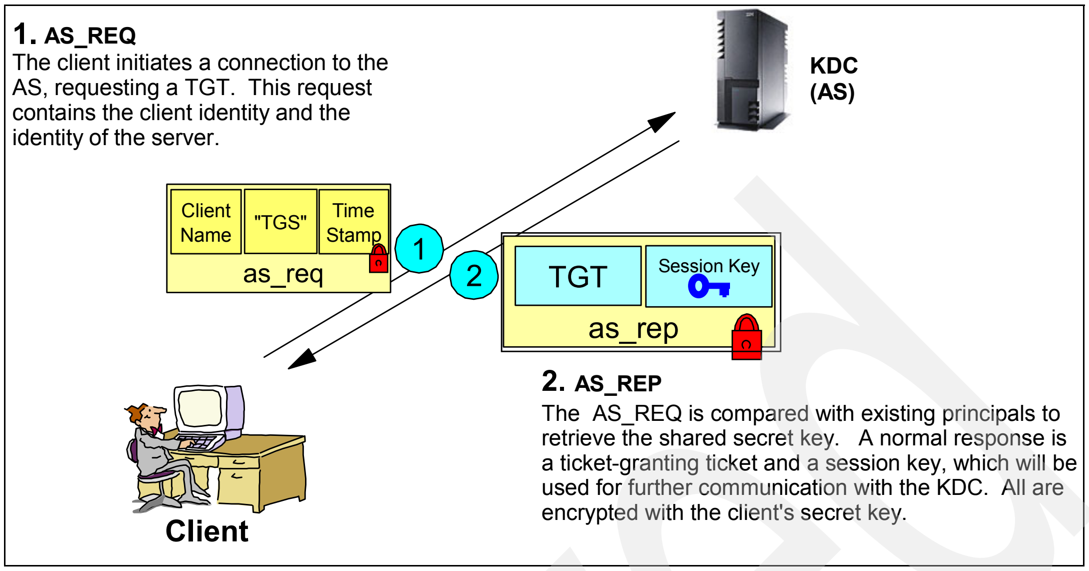

1. AS_REQ
The client initiates a connection to the AS, requesting a TGT. This request contains the client identity and the identity of the server.
KDC (AS)
Client Name
"TGS"
Time Stamp
as_req
1
2
TGT
Session Key
as_rep
2. AS_REP
The AS_REQ is compared with existing principals to retrieve the shared secret key. A normal response is a ticket-granting ticket and a session key, which will be used for further communication with the KDC. All are encrypted with the client's secret key.
Client

Figure 4-2 Receiving a TGT (AS_REQ and AS_REP)

As well as passing back the TGT the AS creates a *session key* so that the principal can communicate with the TGS.

Receiving a Service Ticket

When a principal wants to access a service this TGT is given to the Ticket Granting Server, in addition the principal name of the service that the principal wants to access. This request is known as a TGS_REQ. The TGS takes the TGT and decrypts it using the TGS's shared secret key with the AS, this is done to make sure that the ticket is valid, the session key also becomes available after this decryption. Along with the principal name of the service an *authenticator* is also sent. An authenticator is a piece of information, usually a timestamp, which is encrypted using the session key. The TGS can decrypt this authenticator because of decrypting the TGT and retrieving its half of the session key. The TGS can then check to see if the ticket is valid. It does this by testing to see if the ticket has expired, or if the ticket has been post dated. If the TGT did not have these two levels of encryption then it would be possible for a malicious user to indefinitely extend the life of a ticket. We talk about how Kerberos has built in security to combat situations like these, later on in this chapter.

The TGS then issues the principal a Service Ticket (ST). This reply from the TGS is known as a TGS_REP. The ST allows the principal to access the requested service assuming the principal has proved its identity to the service and that the principal is authorized to use that service. The TGS_REP containing the service ticket is encrypted using the session key created for communication between the principal and the TGS. This is outlined in Figure 4-3.

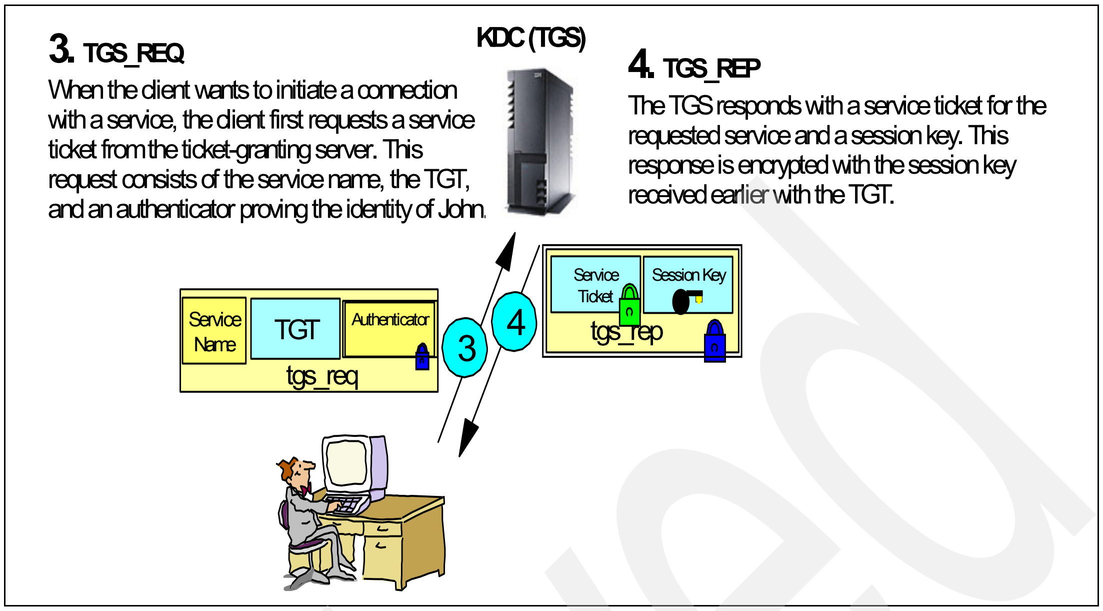

3. TGS_REQ
When the client wants to initiate a connection with a service, the client first requests a service ticket from the ticket-granting server. This request consists of the service name, the TGT, and an authenticator proving the identity of John.
KDC (TGS)
4. TGS_REP
The TGS responds with a service ticket for the requested service and a session key. This response is encrypted with the session key received earlier with the TGT.
Service Name
TGT
Authenticator
tgs_req
3
4
Service Ticket
Session Key
tgs_rep

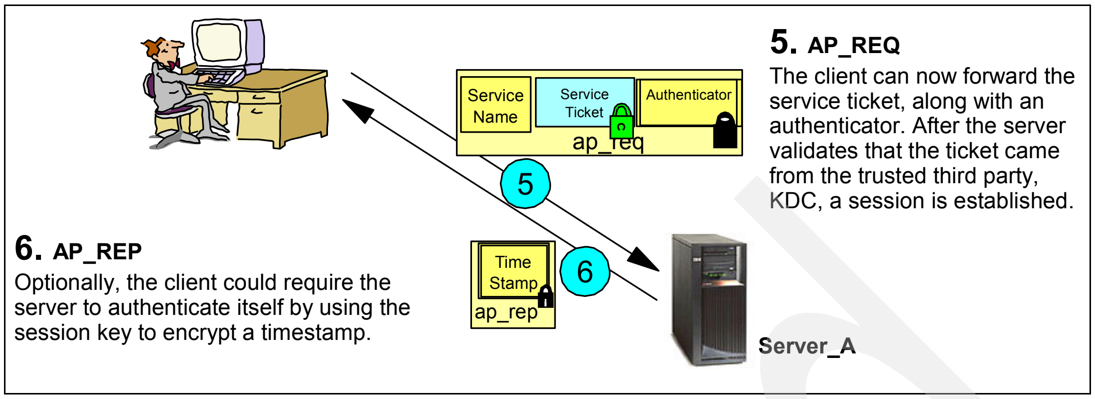
Figure 4-3 Receiving a Service Ticket(TGS_REQ and TGS_REP)

After the ST has been received and decrypted the principal can initiate communications with a service.

Communicating with a kerberized service

The principal now has in its possession a service ticket and a session key. The service ticket allows the principal to talk to the service and the session key is used to encrypt communications and verify the identity of the principal(Figure 4-4).

A AP_REQ now takes place. The principal sends to the service computer, the service name, along with the service ticket and an authenticator. The service ticket like the TGT is encrypted using the shared secret between the AS and the service. The service decrypts this ticket to obtain information about the principal requesting the service and to retrieve the session key that can be used to verify communications with the principal. When the service has the session key it can the decrypt the authenticator. If the authenticator decrypts correctly then the service knows its talking to a valid principal of the service. A AP_REP is then initiated from the service to the principal. This response is an authenticator generated by the service. This authenticator could be fabricated though by replaying the original AP_REQ authenticator sent to the service. To combat this after the authenticator has been decrypted it is returned with the principal name of the service included, this is also done to verify the identity of the service principal to the client principal.


5. AP_REQ
The client can now forward the service ticket, along with an authenticator. After the server validates that the ticket came from the trusted third party, KDC, a session is established.
Service Name
Service Ticket
Authenticator
ap_req
5
6. AP_REP
Optionally, the client could require the server to authenticate itself by using the session key to encrypt a timestamp.
Time Stamp
ap_rep
6
Server_A
Figure 4-4 Accessing the service (AP_REQ and AP_REP)

The Kerberos protocol uses the same procedure for all communications.

4.2.4 Kerberos Security

One of the weaknesses of Kerberos is if a malicious principal want to cause a potential denial of service attack it could replay replies from the AS and the TGS to the client principal. This would have the effect of returning a ticket which could not be decrypted by the user, if the message replayed was not a ticket meant for that principal. It could even have the effect that a ticket which has already expired is sent back to the principal so that no actions can be performed on the network. To combat this a random value called a *nonce* is used. This nonce is sent in requests to the AS and the TGS encrypted. In the AS request it is encrypted with the shared secret of the principal and the KDC, and in the TGS request it is encrypted using the session key returned from the AS reply. The AS or TGS (KDC) receives the request ticket, decrypts it and obtains the nonce. This nonce is then modified, usually by performing an arithmetic operation on it which can be as simple as adding or subtracting one. This modified nonce is then used in replies to the client principal so that potential denial of service attacks can be countered.

In addition to the nonce a *replay cache* is also maintained by any kerberized service. This replay cache does not cache each request, it only caches the authenticator. If the replay cache received a message in which the authenticator is identical to another stored in the cache then it is discarded.

As discussed in “Receiving a Ticket Granting Ticket” on page 40 in Kerberos Version 4 it was possible to request ticket granting ticket which was not intended for you. This ticket could then have an offline brute force operation levied against it to find a principals shared secret. To prevent this in Kerberos 5 a pre-authentication procedure can be implemented in the KDC. If this pre-authentication is turned on it forces the principal to prove its identity before a TGT is issued to that user. Figure 4-5 shows the AS_REQ exchange with the additional pre-authentication information appended on the end of the transfer.

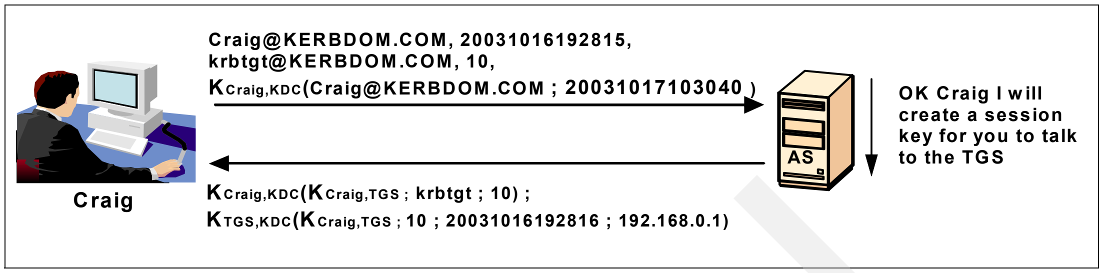

Craig@KERBDOM.COM, 20031016192815,
krbtgt@KERBDOM.COM, 10,
$K_{Craig,KDC}$(Craig@KERBDOM.COM ; 20031017103040)
AS
OK Craig I will create a session key for you to talk to the TGS
Craig
$K_{Craig,KDC}$($K_{Craig,TGS}$; krbtgt ; 10) ;
$K_{TGS,KDC}$($K_{Craig,TGS}$; 10 ; 20031016192816 ; 192.168.0.1)

Figure 4-5 AS_REQ with Pre-Authentication

It is a recommendation that Kerberos networks are time synchronized. When an authenticator is sent in a request to a service, or the KDC, the timestamp contain therein is compared with the time on the local machine. If the times differ by more than the amount allowed by the maximum time defined an error message is returned. By default the maximum time is five (5) minutes. On the iSeries this can be configured up to 900 seconds but for security reasons that is not recommended. See “Time / SNTP” on page 21 for more information, or “Maximum Clock Difference” on page 53 for assistance with configuring this option on the iSeries. This is a measure to help defend against replay attacks.

The Kerberos protocol although very secure is by no means infallible. The protocol may work as it is intended but there are many outside factors which can greatly impact the Kerberos implementation on a network. These are:

- Unauthorized access to a client machine. This could lead to users tickets being copied and a malicious user could impersonate this user until the tickets expire.
- Unauthorized access to servers. This would lead to the compromise of all the services that a server would offer. It could also lead to compromise of kerberos principals also as a higher level of authority could be achieved for users through potential services offered by other machines.
- Compromise of the KDC. If the KDC is compromised then a malicious user would have full control over the entire KDC. The database of all users shared secrets would be compromised also because by default Kerberos implementations allow root users and Kerberos administrators unencrypted access to the database.
- Compromise of users passwords. If a password is divulged to another user by legitimate means or by unauthorized means that user is compromised. This could be especially severe if it is the Kerberos administrators password which is divulged.
- As discussed previously, replay of tickets could prevent users being able to perform actions on the network but if this is taken one step further and replay of request packets is levied against the KDC then this can prevent the replies from authentic users for TGTs or STs.
- Password length and characters contained therein is also a critical issue of security on the network. Administrators should enforce rules that make passwords secure as possible. For example a minimum of number characters, uppercase and lowercase, that must include numeric digits.
- Social engineering attacks make Kerberos vulnerable. For example a user pretends to be an administrator and needs the users password in order to make something on his computer better. It may not even be as devious as that, a friend may even just ask for the password to gain access to something he does not have access to normally.

4.2.5 Kerberos and Microsoft

Kerberos, unknown to many users and administrators, is actually implemented in Microsoft Windows 2000 servers and later. The Kerberos implementation in Windows conforms to the protocol specification as outlined above and in Internet RFCs. There are two parts to each Kerberos implementation on a Windows server. These are the Kerberos service and also Active Directory which is Microsoft's implementation of the LDAP protocol.

When a user logs onto the Windows work domain the information entered by the user is captured by the logon program and is transmitted to the computer's Local Security Authority (LSA). The LSA is a windows component that authenticates users to the local system. The LSA also controls all aspects of local security on a system. This LSA then communicates with the network's KDC in order to receive ticket granting tickets and service tickets so that this user can access kerberized services on the windows domain.

Kerberos on Windows server platforms uses the Active Directory for all information about principals on the Kerberos network. The encryption key used in communicating with principals is stored in the Active Directory database in the user's profile. This password is highly secured and the password object is only an encryption key derived from the password. In addition to this only the person who the account represents can change the password, not even administrators have permissions to do this. Password changes are managed through trusted security programs running in the security context of the LSA. From this we can see that the Active Directory not only plays the role of an LDAP server on a Windows server it is also used as the KDC database, this can be a great benefit to an organization in terms of one central store but can be equally devastating in the event of system failure.

4.2.6 Kerberos commands

In this section there is an overview of some of the more common commands Kerberos principals can use within a realm. When a command is run it runs using the principal name of the logged in user. Here is the list of commands:

- **`kinit`**, this command authenticates the principal with the KDC and retrieves a Ticket Granting Ticket for that user.
- **`klist`**, this command shows a list of all the tickets relating to this principal. The klist command is not installed by default in windows, for information on this command and where to find it, see "Klist command" on page 238.
- **`kdestroy`**, this command destroys the list of tickets that are available to the local principal.
- **`kpasswd`**, this command is used to change the Kerberos password for this principal user. This command requires that the kpasswd (kpasswd daemon) service is running on the server.
- **`kadmin`**, this command can be used to administer principals on the KDC. It does require the kadmind (kadmin daemon) to be running on the computer containing the KDC.

It is worth noting that under the Windows implementation of Kerberos the kinit and kdestroy commands are performed for principals under the covers of the operating system. This is done through the **`kinit`** command which requests TGTs for principals. Similarly under the covers of the operating system **`kdestroy`** is ran when the user logs out of Windows removing the cache of tickets relating to that user.

The commands listed above are the standard Kerberos commands there are however some tools from Windows which run on the command line to allow certain Kerberos functions for interoperability between operating systems and Kerberos implementations. These are:

- **ktpass**, this command is used to allow interoperability between a Windows KDC and non-Windows clients such as UNIX, Linux or iSeries. The command links the name of a principal on the network to the entry in Active Directory and assigns a password to it. As well as linking principals to accounts a UNIX style keytab is also created. This can be transferred to Linux and UNIX machines and is the used by the client to communicate with the Windows server. For the syntax of the ktpass command see “Associate the account with the Kerberos principal” on page 107.
- **ksetup**, command is used to define the realm name, the hostname of the KDC server, and the local machine password.

The support tools are not included in the standard windows installation but are available from the Windows server cd. For information on how to install these support tools, see Appendix C., “Windows 2000 Kerberos tools” on page 235. This appendix also gives links to some useful tools which can be used to help users and administrators look at the tickets on a a system and perform helpful functions on these tickets.

The commands listed in this section are executable from the command line in either Windows or Linux Kerberos implementations. The commands are also available to be run in a QShell on the iSeries, for an example of these commands in a QShell and the relevant output see “Kerberos Client tasks through Qshell Interpreter” on page 64.

4.3 Kerberos summary

There are firewalls to protect our networks from intrusion from the outside but many attacks come from within our own network. Kerberos has many good intranet security features. It provides encryption for passwords on the network and also reduces the need to use passwords. Kerberos can be implemented on most platforms and is readily accessible for download.

Advantages of Kerberos:

- Passwords are never transmitted across the network and so are not vulnerable to malicious users eavesdropping on the line.
- Single signon can be achieved through Kerberos.
- Due to in built security Kerberos tickets are only useful for their intended recipients and it is very hard to either forge a ticket or impersonate a user on a Kerberos network if user passwords are secure.

Disadvantages of Kerberos:

- To be used as a single signon solution, users must synchronize their IDs.
- The Key Distribution Center without any slave KDCs is a single point of failure on the network.
- Kerberos does not prevent denial of service or social engineering attacks.
- The Kerberos network is only as secure as the physical security of the actual computers on the network, especially the computer with the KDC.

Although Kerberos is implemented on networks with a Windows 2000 server and later most administrators are unaware of its existence and that it is used under the covers of the operating system each time a user logs on to his computer. There are disadvantages to Kerberos but they are not just Kerberos related. Most of these disadvantages are actual company security concerns such as the physical access to servers and who should be allowed to do so by the companies security policy. The network disadvantages are listed here to remind you that these points are the underlying security issues that an organization has to solve and although implementing kerberos will make you electronic network more secure it does nothing for the physical implementation.

4.3.1 Where to obtain Kerberos

If you want to download an implementation of Kerberos to implement other than the default Windows Kerberos version, follow the links below.

MIT Kerberos is available from:

`http://web.mit.edu/Kerberos/`

Heimdal Kerberos i s available from:

`http://www.pdc.kth.se/heimdal/`

5

iSeries Network Authentication Service

Network Authentication Service allows the iSeries, and applications running on the iSeries, to accept Kerberos tickets as a means of authenticating principals. With Network Authentication Service configured to the iSeries it can act as a client using Kerberos tickets to authenticate itself to a kerberized application. You will recall from 4.2.2, “Principals and realms” on page 25, that a principal is a member of a Kerberos realm and can represent a person, server, or a service. Network Authentication Service also provides the link that binds Kerberos authentication with the IBM @server EIM to implement a complete single signon (SSO) solution.

Network Authentication Service first became available on the iSeries at V4R5 with program temporary fix (PTF) SF63662. At V5R1 Network Authentication Service was included in the base operating system of the iSeries. The two releases prior to V5R2 require that you manually configure Network Authentication Service. This book is based on V5R2 and later releases to demonstrate the use of iSeries Navigator’s tools to configure Network Authentication Service.

The iSeries uses the following standards to implement Network Authentication Service:

- Kerberos
  - Designed by MIT.
  - Third-party distributed authentication protocol suitable for heterogeneous environments.
  - Kerberos Version 5 protocol described in Request For Comment (RFC) 1510.
  - Many of the de facto standard Kerberos protocol APIs that are prevalent in the industry today.
- Generic Security Service (GSS) Application Programming Interface (API)
  - An API set for secure exchange of information between applications.
  - GSS APIs are defined in RFCs 1509,1964, and 2078.

In this chapter we discuss these topics:

- 5.1, “Managing Network Authentication Service” on page 50
- 5.2, “Administrative tasks in iSeries Navigator” on page 59
- 5.3, “Kerberos Client tasks through Qshell Interpreter” on page 64
- 5.4, “More information” on page 70

5.1 Managing Network Authentication Service

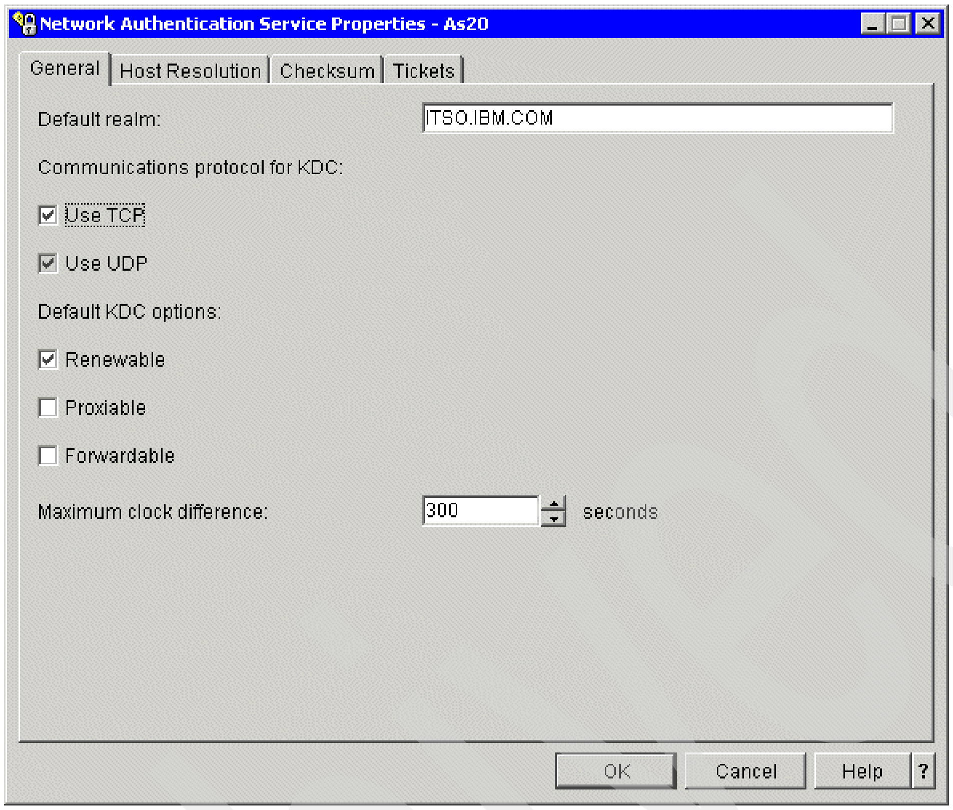
After configuring Network Authentication Service on the iSeries, which is described in Chapter 7, “Enabling Network Authentication Service and Enterprise Identity Mapping” on page 51 you must maintain the environment. Kerberos ticket management can be accomplished using the QShell interpreter’s command line interface. Managing the network configuration is done either through the iSeries Navigator’s Graphical User Interface (GUI), or by manually editing the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file. We focus on making the changes in the GUI interface, but will show the resulting changes in the krb5.conf file in a shaded note box.

The iSeries Navigator interface can be used to administer the Network Authentication Service environment. To use the GUI you must first meet the requirements listed in 5.2, “Getting started” on page 40.

1. To get to the properties window via iSeries Navigator you first need to sign on to the iSeries.
   a. To bring up the signon window left-click the system name in the left pane of the iSeries Navigator interface.
   b. When prompted enter the user name and password to the system. Remember to use a profile that has *SECADM, *ALLOBJ, and *IOSYSCFG authority.
2. To display or change the properties for Network Authentication Service in iSeries Navigator you must navigate to the Network Authentication Service Properties window.
   a. In the navigation pane select your **iSeries -> Security**.
   b. Right-click **Network Authentication Service** and select **Properties**.

5.1.1 Parameters in the General window

Parameters in the General window shown in Figure 5-1,define the default realm, the communications protocol to be used and several of the definable ticket options.


Network Authentication Service Properties - As20

General | Host Resolution | Checksum | Tickets

Default realm: ITSO.IBM.COM

Communications protocol for KDC:

[✓] Use TCP

[✓] Use UDP

Default KDC options:

[✓] Renewable

[ ] Proxiable

[ ] Forwardable

Maximum clock difference: 300 seconds

Figure 5-1 Parameters in the General window

Default realm

A Kerberos realm is the set of principals that have a common Key Distribution Center (KDC). The default realm is the KDC that the iSeries will initially contact to authenticate. Kerberos conventions specify that the default realm is listed in upper case. In this case the default realm is ITSO.IBM.COM, as shown in Figure 5-1. This can be compared to the configuration file on the iSeries by running the edit file command:

edtf '/QIBM/UserData/OS400/NetworkAuthentication/krb5.conf'

Note: In the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file you will find an entry that contains the name of the default realm:

```
default_realm = ITSO.IBM.COM
```

Communications protocol for KDC

The default communications protocol for Kerberos is User Datagram Protocol (UDP). In an environment where all of the transactions between the principal and the KDC can be handled in a single packet UDP is sufficient. However when Windows 2000 systems are present on the network the datagrams can be significantly larger than the maximum frame size of 1500 bytes allowed on many ethernet networks. Depending on the devices in your network it may be more reliable to use Transmission Control Protocol (TCP) connections. Even if the Use TCP option is selected as in Figure 5-1, the Use UDP option remains checked. If the TCP connection cannot be established the system will retry to connect using UDP.

Note: When the **Use TCP** option is selected the following entry is made in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

```
kdc_use_tcp = 1
```

Default KDC options

When you request a Ticket Granting Ticket (TGT) from the KDC there are several options available that determine how that ticket can be used later. The default KDC options that are available through iSeries Navigator are renewable, proxiable, and forwardable. Each of these options correspond with a flag in the TGT. Many of the flags are requested and set when the initial request for the TGT is made by the principal. Other flags are configured on the server and cannot be changed regardless of what the client requests. A complete list and description of all the flags can be found in Section 2 of RFC 1510 which can be referenced on the Internet at:

http://www.faqs.org/rfcs/rfc1510.html

Figure 5-1 shows the options that you can select when requesting tickets.

Renewable

The renewable option allows a principal, usually an application or service, to request a ticket that is usable for an extended period of time. Renewable tickets contain two expiration times. The first expiration time represents the maximum lifetime the current ticket can be used. The second timestamp sets the latest time that the ticket may be renewed. If the principal requests that the ticket be renewed, prior to the expiration of the maximum lifetime timestamp, the ticket will be renewed until for another maximum lifetime interval. This can be repeated until the second expiration time is reached.

Proxiable

Setting the proxiable flag on the TGT request allows you to request a service ticket from an Internet Protocol (IP) address other than the IP address the TGT was originally requested from. The service ticket is then allowed to perform a task on behalf of the principal. Since the service ticket that was issued from the original TGT was already authenticated no password is required.

Forwardable

Forwardable tickets are very similar to proxiable tickets except that they can also include TGTs. When the forwardable flag is turned on a copy of the TGT is sent to the target system as part of the logon process. That system can in turn request service tickets using its own IP address. Once again, no password is required on any of the remote systems.

Note: The following entries are made in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file depending on the options checked:

`kdc_default_options = 0x00000000` = No options selected

`kdc_default_options = 0x00000010` = Renewable option only

`kdc_default_options = 0x10000000` = Proxiable option only

`kdc_default_options = 0x10000010` = Both the proxiable and the renewable options

`kdc_default_options = 0x40000000` = Forwarding option only

`kdc_default_options = 0x50000000` = Both proxiable and forwarding options

`kdc_default_options = 0x50000010` = All default KDC options

Maximum Clock Difference

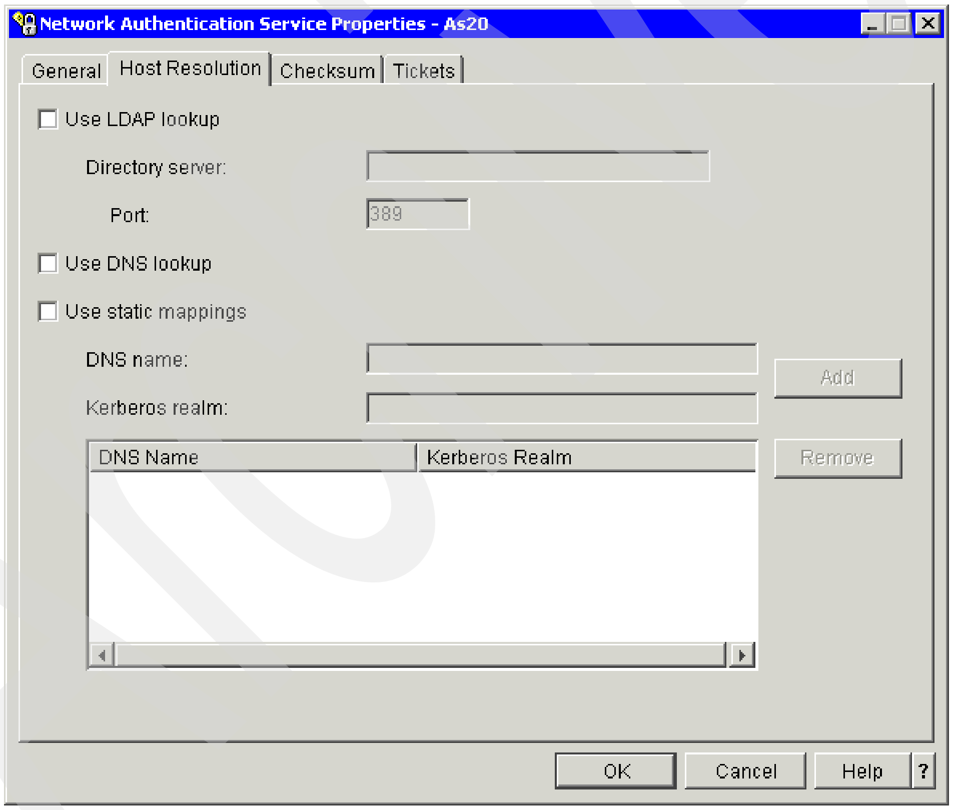
When a principal requests a ticket from the KDC there is a timestamp in the request. The maximum clock difference parameter reflects the maximum amount of time, in seconds, that the KDC's clock and the timestamp in the request can differ. If the difference exceeds the specified value the request is rejected. The valid range for the parameter is 0 to 900 seconds. Figure 5-1 shows the clockskew parameter with the default value of 300 seconds.

Note: When a value other than 300 seconds is specified the following entry is placed in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

```
clockskew = 400
```

5.1.2 Parameters on the Host Resolution window

The Host Resolution tab shown in Figure 5-2, provides you with the methods to configure how you resolve host and realm names. There are three choices in the window, LDAP Lookup, DNS Lookup and Static mappings. Selections in this window are optional, you may choose not to select any or you may select any combination the choices. If more than one is selected, the order of search precedence is LDAP lookup, then DNS lookup and finally static entries.



☐ Use LDAP lookup

Directory server:

Port: 389

☐ Use DNS lookup

☐ Use static mappings

DNS name:

Kerberos realm:

DNS Name	Kerberos Realm

Figure 5-2 Parameters in the Host Resolution window

Use LDAP Lookup

If you are using Lightweight Directory Access Protocol (LDAP) to resolve host names select the *Use LDAP Lookup* option and enter the name of the LDAP server. By default LDAP listens on port 389 which is already supplied in the port parameter. If your LDAP server does not listen on the default port you may specify the correct port for your server. Valid port numbers are 1-65535.

Note: The following entries are placed in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file when the Use LDAP Lookup is selected:

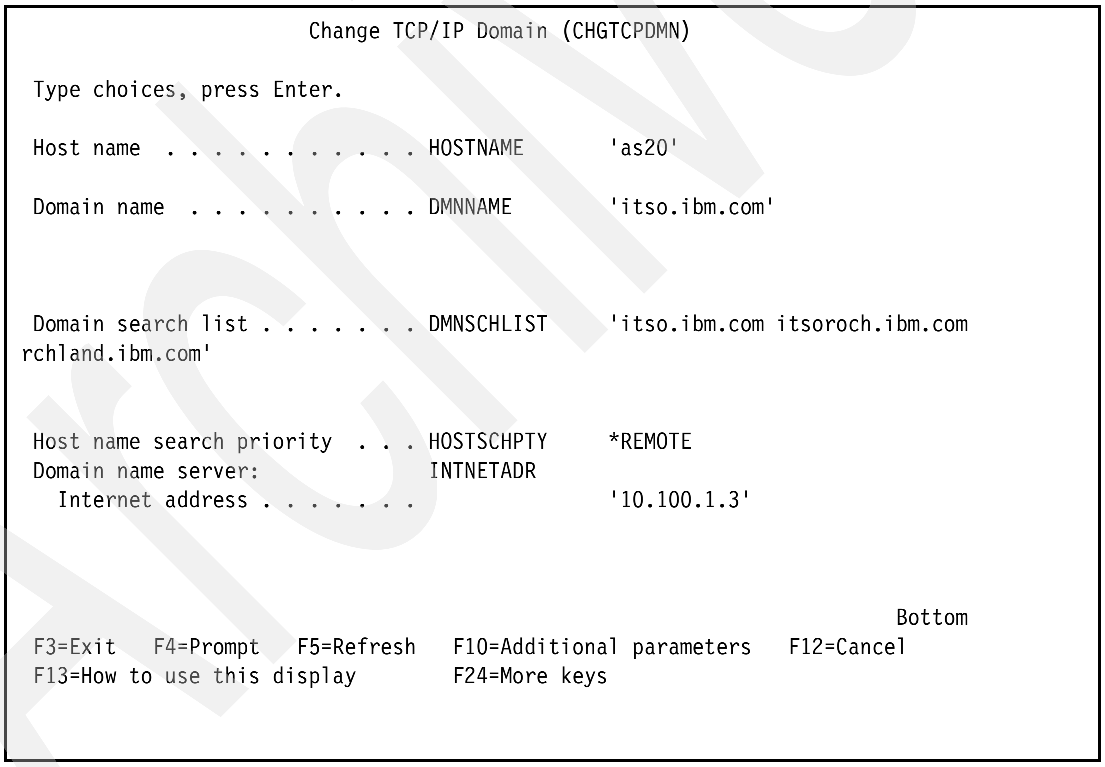
```
ldap_server = YourServer.YourDomain.com:389
use_ldap_lookup = 1
```

Use DNS Lookup

If you select the *Use DNS lookup* option host resolution is performed by the DNS server that is listed in the domain name server (INTNETADR) parameter of the CHANGE TCP/IP DOMAIN (CHGTCPDMN) command. In the CHGTCPDMN screen (Figure 5-3) when the Host name search priority (HOSTSCHPTY) is set to *LOCAL the iSeries' host table will be searched for a matching entry. If no entry for the specific host is found the request will then be forwarded to the DNS server to be resolved.

Note: When you select the Use DNS lookup the following entry in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file is made:

```
use_dns_lookup = 1
```



```
                         Change TCP/IP Domain (CHGTCPDMN)

 Type choices, press Enter.

 Host name  . . . . . . . . . . . HOSTNAME       'as20'

 Domain name  . . . . . . . . . . DMNNAME        'itso.ibm.com'



 Domain search list . . . . . . . DMNSCHLIST     'itso.ibm.com itsoroch.ibm.com
rchland.ibm.com'


 Host name search priority  . . . HOSTSCHPTY     *REMOTE
 Domain name server:              INTNETADR
   Internet address . . . . . . .                '10.100.1.3'



                                                                        Bottom
 F3=Exit   F4=Prompt   F5=Refresh   F10=Additional parameters   F12=Cancel
 F13=How to use this display        F24=More keys
```

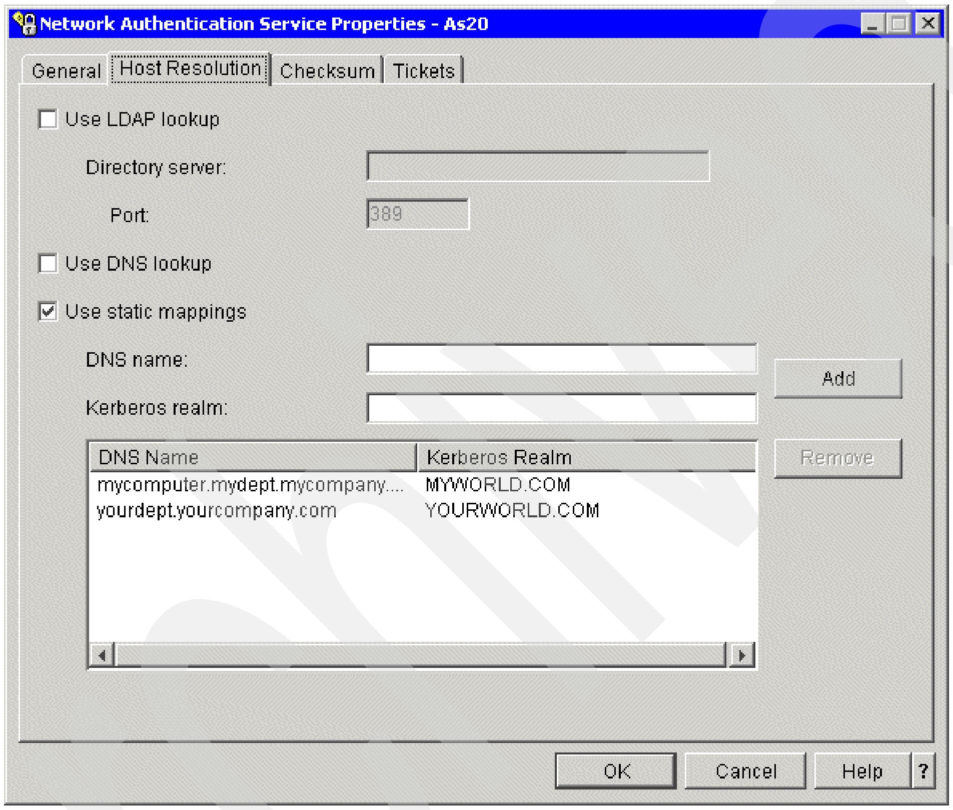
*Figure 5-3 Change TCP/IP Domain with the Host name search priority set to *REMOTE*

Use Static Mappings

You can also choose *static mapping* as a method of resolving host names on your network, as shown in Figure 5-4. With this option you can use either specific host names or generic host names. An example of a specific host name, if your computer is named my computer, would be mycomputer.mydept.mycompany.com with a Kerberos realm of MYWORLD.COM. Generic entries can be used to map entire domains to a realm. For example, if the fully qualified name of your workstation is yourcomputer.yourdept.yourcompany.com. That host name could be mapped to the Kerberos realm YOURWORLD.COM with a static mapping of yourdept.yourcompany.com in the Kerberos realm YOURWORLD.COM. That entry would associate all of the hosts in yourdept.yourcompany.com with YOURWORLD.COM.

Note: The entries in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file look like this when you add static mappings:

```
mycomputer.mydept.mycompany.com = MYWORLD.COM
yourdept.yourcompany.com = YOURWORLD.COM
```



☐ Use LDAP lookup
Directory server:
Port: 389
☐ Use DNS lookup
☑ Use static mappings
DNS name:
Kerberos realm:

DNS Name	Kerberos Realm
mycomputer.mydept.mycompany....	MYWORLD.COM
yourdept.yourcompany.com	YOURWORLD.COM

Figure 5-4 Static Mapping entries

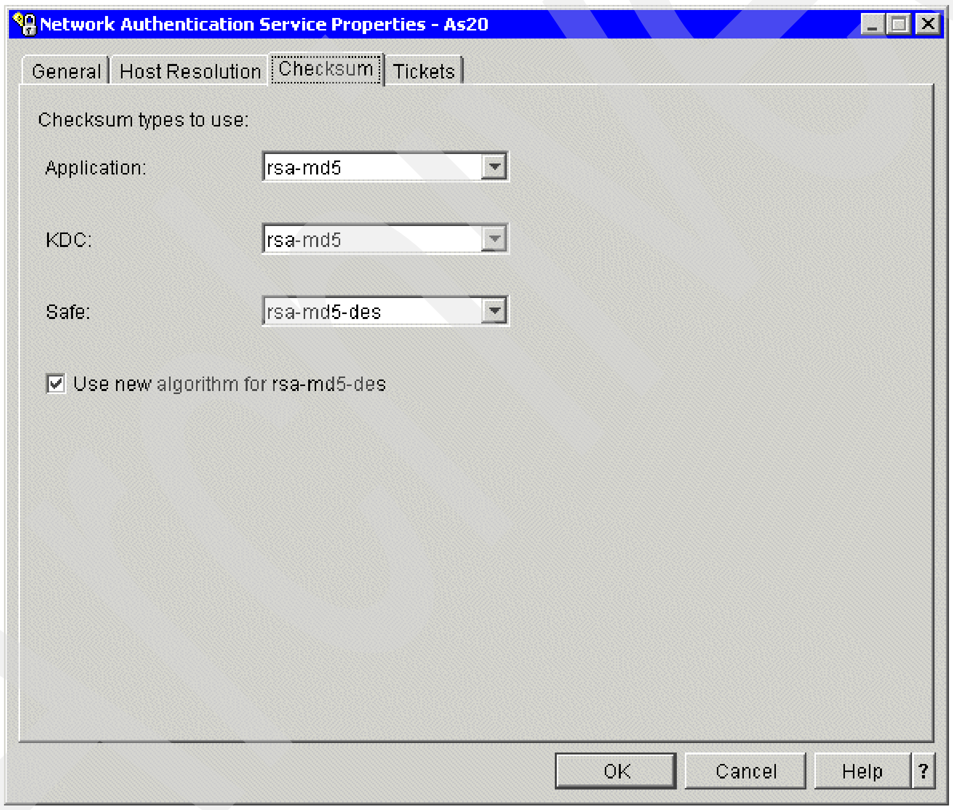
5.1.3 Parameters on the Checksum window

Kerberos uses checksums to insure message integrity. Checksums provide some level of assurance that the messages have not been altered between the time they were sent and the time they were received. Both of the principals in the conversation have a key to encrypt and decrypt the messages they exchange. Prior to the data being sent it is submitted to a mathematical algorithm that produces the checksum value. That value is then attached to the message and sent. On the receiving system the content of the message is run through the same algorithm again. When the checksum still calculates to the same value it is assumed that the data has been not been altered. If the checksum value has changed the message is rejected. Although using a checksum provides message integrity, it does not securely encrypt the data. This means that the end user can have some level of assurance that the message has not been altered, but it does not mean that the message has not been observed on the network.

The checksum options provided on the iSeries are:

- crc32 - Calculates a 32-bit checksum using a cyclical redundancy check algorithm. This checksum is vulnerable to some forms of attacks and is not guaranteed to be collision proof.
- des-cbc - Calculates a 64-bit checksum using DES encryption in CBC mode.
- rsa-md5 - Calculates a 128-bit checksum using the rsa-md5 algorithm.
- rsa-md5-des - Calculates a 128-bit checksum using the rsa-md5 algorithm and then encrypts the checksum using DES encryption in CBC mode.
- nist-sha - Calculates a 160-bit checksum using the NIST (National Institute of Standards and Technology) SHA-1 algorithm.
- hmac-sha1-des3 - Calculates a 160-bit checksum using the NIST SHA-1 algorithm and then encrypts the checksum using Triple-DES encryption in CBC mode.

In the Checksum window (Figure 5-5) you can change the checksum algorithm used for the application, the KDC and the safe messages.



Checksum types to use:

Application: rsa-md5

KDC: rsa-md5

Safe: rsa-md5-des

☑ Use new algorithm for rsa-md5-des

Figure 5-5 Parameters in the Checksum window

The Application parameter

The Application parameter refers to any application running on the system you are currently configuring. When an application on this system communicates with another system the checksum specified in this parameter will be used. Make sure that the checksum that is selected on your local system matches the checksum that is used on the remote system. The default value for the Application parameter is rsa-md5.

Note: Changing the application checksum parameter to a value other than rsa-md5 creates the following entry in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

`ap_req_checksum_type = rsa-md5-des` (when the rsa-md5-des type is selected)

The KDC parameter

The value specified in the KDC parameter indicates the checksum that will be used when communicating with the KDC. Make sure that the selection you make on this parameter matches the configuration for the KDC. The default checksum for the KDC parameter is rsa-md5.

Note: Changing the KDC checksum parameter to a value other than rsa-md5 creates the following entry in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

`kdc_req_checksum_type = rsa-md5-des` (when the rsa-md5-des type is selected)

The Safe parameter

Safe messages are sent by applications to verify that a message was received and the checksum value in the message matched the recalculated value. The Safe parameter sets the checksum algorithm to be used on the messages that confirm delivery of valid, or unchanged messages. Following the Safe parameter there is a check box labeled *Use new algorithm for rsa-md5-des*. In Version 5 of Kerberos the rsa-md5-des algorithm was miscalculated. Placing a checkmark in this box insures that the corrected algorithm will be used. The box is checked by default.The default value is rsa-md5-des.

Note: Changing the Safe checksum parameter to a value other than rsa-md5-des (the default) creates the following entry in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

`safe_checksum_type = hmac-sha1-des3` (when hmac-sha1-des3 is selected)

The following entry would indicate that the incorrect algorithm is being used to calculate the checksum

```
rsa_md5_des_compat =  1
```

5.1.4 Parameters on the Tickets window

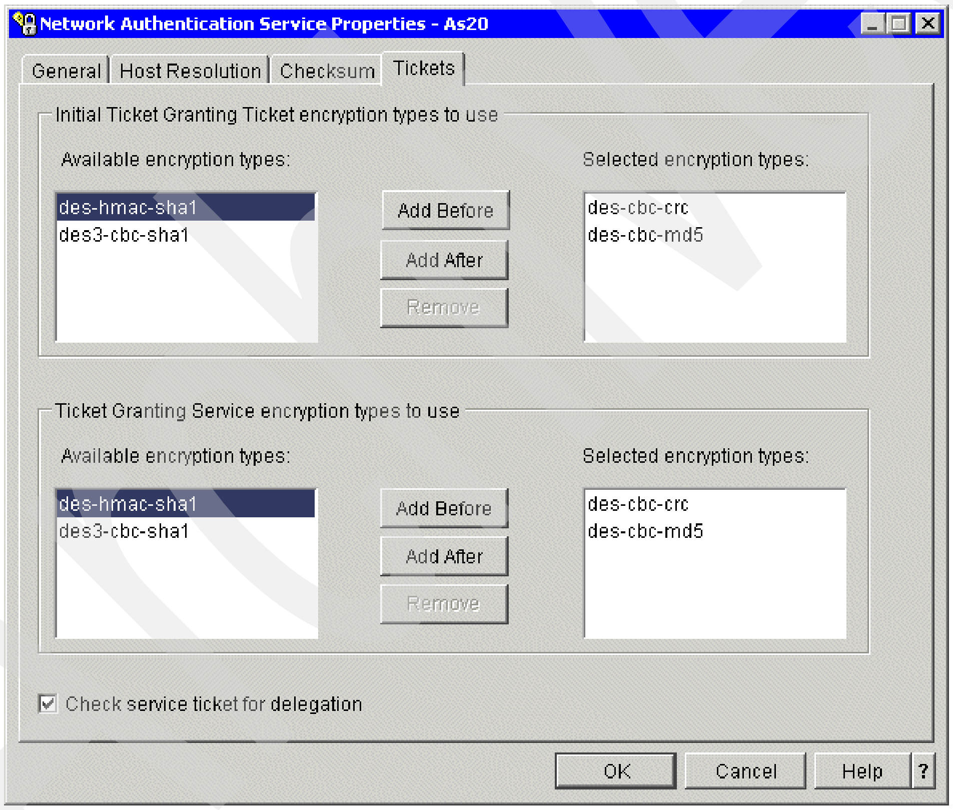
The Tickets tab provides the available options for selecting the encryption levels of ticket requests. There are four encryption levels for Network Authentication Service available on the iSeries:

- des-cbc-crc - Secures tickets with Data Encryption Standard using cipher block chaining and a CRC-32 checksum. Although this encryption type is vulnerable to certain types of attacks, it is supported by all other Kerberos implementations.
- des-cbc-md5 - Secures tickets with Data Encryption Standard using cipher block chaining and an MD5 checksum.
- des-hmac-sha1 - Secures tickets with Data Encryption Standard using cipher block chaining and an HMAC-SHA1 checksum. In addition, key derivation is used to generate unique encryption keys based upon the message type.
- des3-cbc-sha1 - Secures tickets with triple-Data Encryption Standard with cipher block chaining method to encrypt ticket data and generates a hashed message digest for the

checksum. Triple-DES requires more CPU than DES, but it is more secure. Use the hmac-sha1-des3 checksum value with this encryption type.

For ticket requests to be encrypted at least one of the encryption type(s) selected must be supported on the KDC. The type of encryption that is selected on the Tickets tab is applied to the client not the KDC. When a ticket request is made the KDC compares its own list of encryption types to the list of encryption types supported by the client. The KDC then selects the most secure means available to both principal and KDC to secure the connection. When you make the decision about the level of encryption to use you need to find the proper balance between the amount of security necessary on your network and the additional overhead that is required to encrypt messages.

The two primary sections on the Tickets window, Initial Ticket Granting Ticket encryption types to use, and Ticket Granting Service encryption types to use, are very similar in appearance. The difference between the two is that the first section allows you to select encryption types to be used by the client when requesting the TGT. The second section applies to the encryption level used by the client when requesting service tickets from the ticket granting service (TGS). The two sections can be seen in Figure 5-6.


Network Authentication Service Properties - As20

General | Host Resolution | Checksum | Tickets

Initial Ticket Granting Ticket encryption types to use

Available encryption types: des-hmac-sha1, des3-cbc-sha1

Selected encryption types: des-cbc-crc, des-cbc-md5

Ticket Granting Service encryption types to use

Available encryption types: des-hmac-sha1, des3-cbc-sha1

Selected encryption types: des-cbc-crc, des-cbc-md5

☑ Check service ticket for delegation

Figure 5-6 Select client encryption types

The Available encryption types list shows you the types of encryption that have not been selected but are available. Encryption types in the Selected encryption types list display, in order of preference, the types of encryption that can be used by the client when requesting tickets. To add encryption types to be used, first highlight the type that you would like to add. Then, if more than one encryption type exists in the Selected encryption types window, highlight one of them. Finally select either Add Before, or Add After, to move the new type to Selected encryption types list. The KDC will negotiate the type of encryption by attempting each type sequentially until it finds a type it supports. If you want to remove an encryption type from the Selected encryption type list, select the type by clicking on it. After clicking on the type to be removed from the list the Remove button is activated. Clicking on the Remove button will move the selected encryption type to the Available encryption type list.

Note: Adding des-hmac-sha1 between the default entries in the Initial Ticket Granting Ticket and Ticket Granting Service encryption types to use list creates the following changes in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

```
default_tkt_enctypes = des-cbc-crc,des-hmac-sha1,des-cbc-md5
default_tgs_enctypes = des-cbc-crc,des-hmac-sha1,des-cbc-md5
```

The last parameter on the Tickets tab is Check service ticket for delegation. A selecting this option indicates that the server should check the service ticket to confirm whether it may act on someone else's behalf. An example of delegation would be the proxiable, or forwardable, tickets.

Note: Unchecking the box next to Check service ticket for delegation will make the following entry in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

```
check_delegate = 0
```

5.2 Administrative tasks in iSeries Navigator

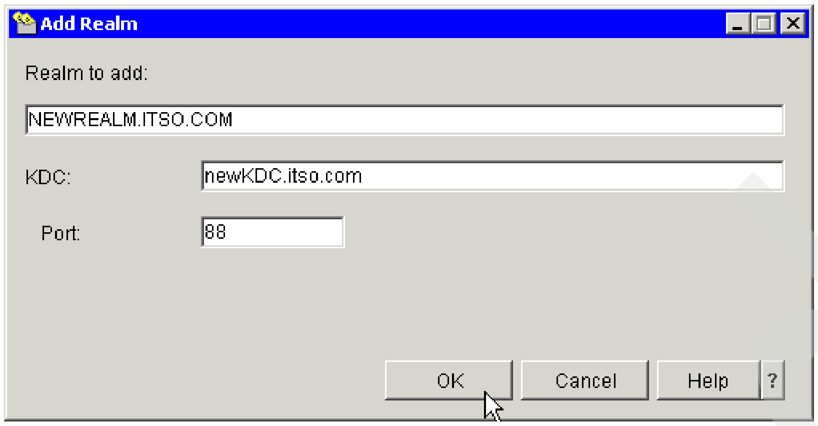
Another facet of managing Network Authentication Service is keeping the realm information up to date. Your configuration needs to be updated when you add or remove realms from you environment. You may also need to change, add, or remove KDCs and password servers in your network. Managing realms, KDCs, password servers and cross realm trusts can all be accomplished through iSeries Navigator. To navigate to Realm Properties start by expanding your **iSeries -> Security -> Network Authentication Service** and then highlight **Realm**.

5.2.1 Adding a realm

1. To add a realm you right-click **Realm** and select **Add Realm**, and the window shown in Figure 5-7 will open.
2. In the Realm to add field enter the name of the realm you are adding. Make sure that you enter the new realm's name in upper case.
3. In the KDC field enter the host name of the new Key Distribution Center.
4. The default port for the KDC will be port 88. If your KDC listens on a different port you may use any port number between 1 and 65535.
5. Click OK.

Note: Adding a new realm creates the following entry in the /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf file:

```
NEWREALM.ITSO.COM = ??<
   kdc = newKDC.itso.com:88
 ??>
```


Add Realm

Realm to add: NEWREALM.ITSO.COM

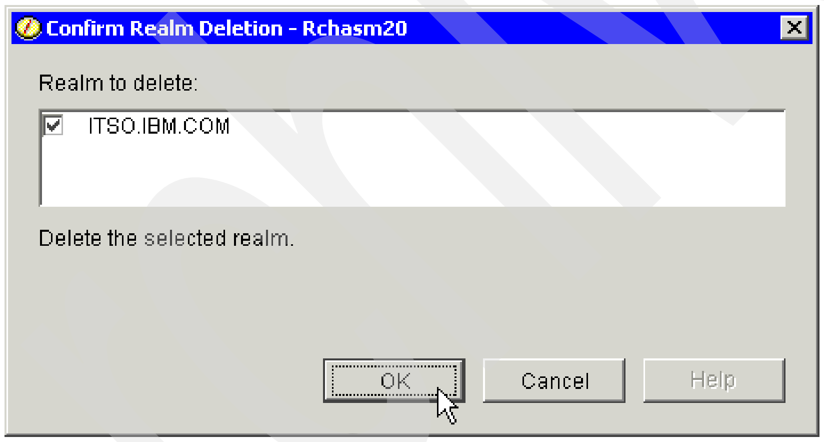
KDC: newKDC.itso.com

Port: 88

Figure 5-7 Adding a new realm

5.2.2 Deleting a Realm

If you want to remove a realm left-click **Realm** in the navigation pane. The list of all realms that have been configured will then appear in the right pane. Right-click the realm that you want to remove and select **Delete**. The Confirm Realm Deletion window then opens (Figure 5-8) that allows you to confirm the realm(s) that you want to remove.


Confirm Realm Deletion - Rchasm20

Realm to delete:

☑ ITSO.IBM.COM

Delete the selected realm.

Figure 5-8 Deleting a realm

5.2.3 Adding and Removing Key Distribution Centers

You can add, remove, or reorder the KDCs listed in your realm using the General tab in iSeries Navigator, Figure 5-9.

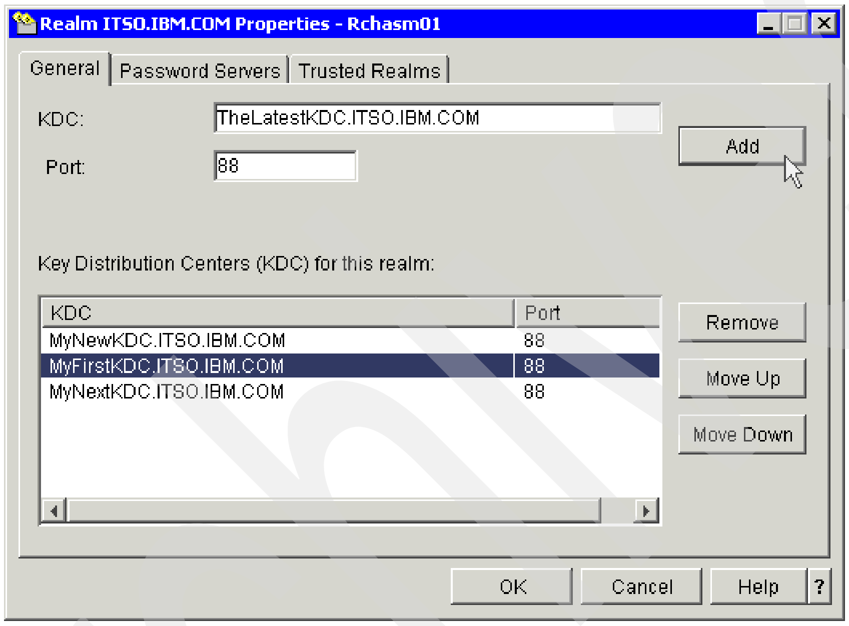
To add KDCs to your realm:

1. In iSeries Navigator Expand your **iSeries -> Security -> Network Authentication Service -> Realm**.
2. In the right pane, right-click the realm that you want to add the new KDC to and select **Properties**.
3. The window opens to the General tab. In the KDC field add the fully qualified host name of the KDC.

4. In the Port field enter the port that the KDC is configured to listen on. The default port is 88 but this could be any port from 1 - 65535.
5. Click **Add**. The name and port you just added will appear in the list of KDCs.
6. Click **OK**.

You can remove KDCs from the realm or reorder the them while in this window also:

1. Highlight the KDC you want to remove and click **Remove**.
2. At this point you may rearrange the order the KDCs are accessed by clicking on the KDC and then clicking on the **Move Up** or **Move Down** buttons.
3. When you have the list ordered the way you want, click **OK**.



KDC: TheLatestKDC.ITSO.IBM.COM

Port: 88

Key Distribution Centers (KDC) for this realm:

KDC	Port
MyNewKDC.ITSO.IBM.COM	88
MyFirstKDC.ITSO.IBM.COM	88
MyNextKDC.ITSO.IBM.COM	88

Figure 5-9 Adding Key Distribution Centers to the realm

Note: Placing additional KDCs in the krb5.conf file creates these entries:

```
ITSO.IBM.COM = ??<
   kdc = MyNewKDC.ITSO.IBM.COM:88
   kdc = MyFirstKDC.ITSO.IBM.COM:88
   kdc = MyNextKDC.ITSO.IBM.COM:88
   kdc = TheLatestKDC.ITSO.IBM.COM:88
   kpasswd_server = p23bk58a.ITSO.IBM.COM:464
 ??>
```

5.2.4 Adding and Removing Password Servers

Password servers allow Kerberos principals to change their passwords. You can add, remove, and reorder the sequence of password servers within the realms through iSeries Navigator, Figure 5-10.

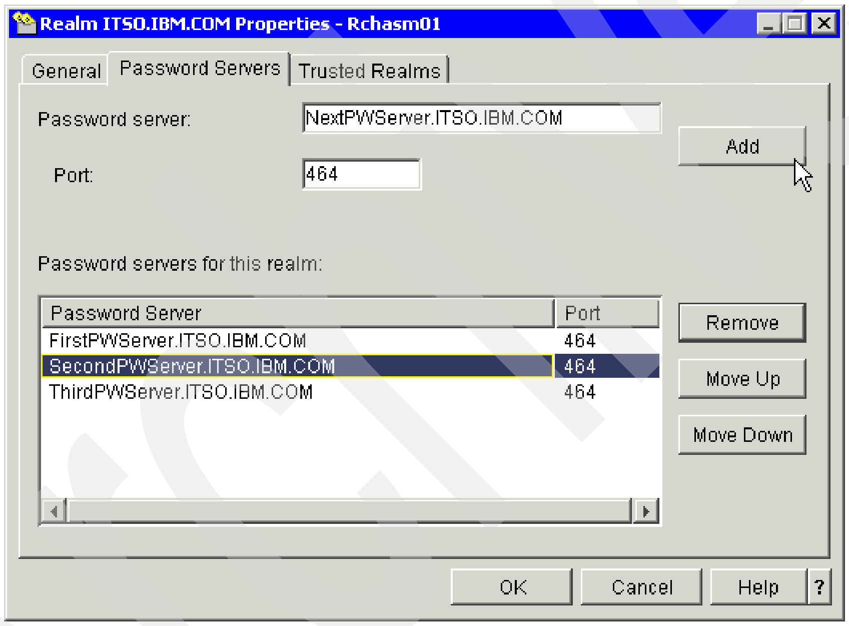
To add a password server:

1. In iSeries Navigator, expand your **iSeries -> Security -> Network Authentication Service -> Realm**.

2. In the right pane, right-click the realm that you want to add the new password server to and select **Properties**.
3. Click the **Password Server** tab. In the Password Server field add the fully qualified host name of the new password server.
4. In the Port field enter the port that the password server is configured to listen on. The default port is 464 but this could be any port from 1 - 65535.
5. Click **Add**. The name and port you just added will appear in the list of password servers.
6. Click **OK**.

You can remove password servers from the realm or change the order of priority while in this window also:

1. Highlight the password server you want to remove and click **Remove**.
2. At this point you may rearrange the order the password servers are accessed by clicking on the password server and then clicking on the **Move Up** or **Move Down** buttons.
3. When you have the list ordered the way you want, click **OK**.



Password server: NextPWServer.ITSO.IBM.COM

Port: 464

Password servers for this realm:

Password Server	Port
FirstPWServer.ITSO.IBM.COM	464
SecondPWServer.ITSO.IBM.COM	464
ThirdPWServer.ITSO.IBM.COM	464

Figure 5-10 Adding and Removing Password servers

Note: Additional password servers create these entries in the krb5.conf file

```
ITSO.IBM.COM = ??<
   kdc = p23bk58a.ITSO.IBM.COM:88
   kpasswd_server = FirstPWServer.ITSO.IBM.COM:464
   kpasswd_server = SecondPWServer.ITSO.IBM.COM:464
   kpasswd_server = ThirdPWServer.ITSO.IBM.COM:464
 ??>
```

5.2.5 Creating and removing cross realm trusts

By creating a cross realm trust relationship you allow principals from other realms to request service tickets for the local system. These service tickets are actually issued by the KDC from the remote realm. The trust relationship between the realms is based on the exchange of shared keys between the local and remote KDCs. Because of the relationship each of the KDCs are able to distribute tickets that will be honored in the other realm. The KDCs must exchange keys before entries are made in this window, Figure 5-11. In the Trusted Realm window you to define “paths” along which cross realm trust has been enabled.

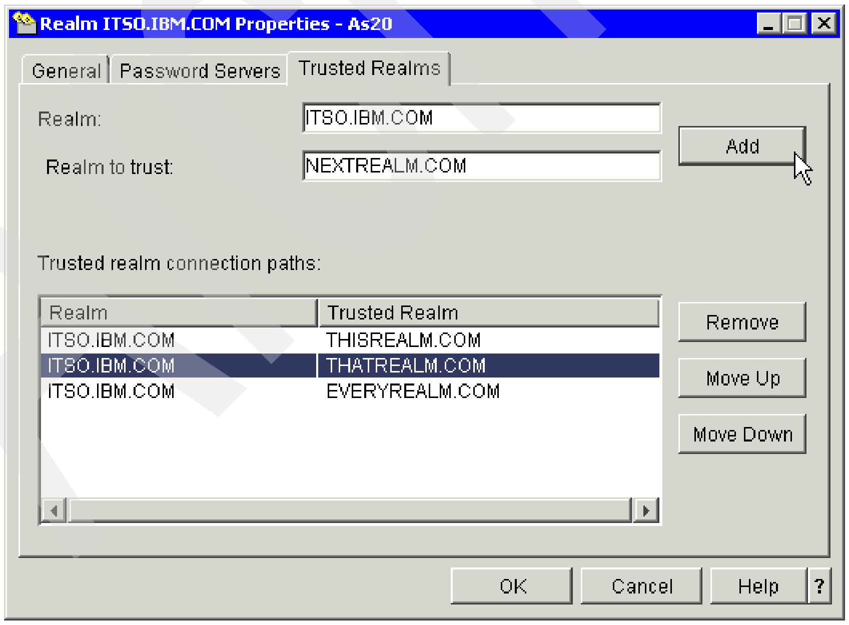
Note: Creating the cross realm trust requires more than just defining the path on the iSeries. There must be an exchange of keys performed on the KDCs.

To add a realm to be trusted:

1. In iSeries Navigator, expand your **iSeries -> Security -> Network Authentication Service -> Realm**.
2. In the right pane, right-click the realm that you want to work with and select **Properties**.
3. Click the **Trusted Realm** tab. In the Realm field add the name of your current realm. Keep in mind that realms need to be entered in uppercase.
4. In the Realm to trust field type the name of the realm that you want to add to the trusted list and then click the **Add** button.You should see the name of the realm added to the Trusted realm connection paths window.
5. Click OK to update the configuration file.

You are also able to remove and rearrange the order of the trusted realms from this same window:

1. To remove a trusted realm, click the name of the realm in the Trusted realm connection paths and select **Remove**.
2. You can change the order of the list by clicking on the name of the realm and then click either the **Move Up** or **Move Down** button.
3. When you have the list ordered the way you want, click **OK**.



Realm: ITSO.IBM.COM

Realm to trust: NEXTREALM.COM

Trusted realm connection paths:

Realm	Trusted Realm
ITSO.IBM.COM	THISREALM.COM
ITSO.IBM.COM	THATREALM.COM
ITSO.IBM.COM	EVERYREALM.COM

Figure 5-11 Adding and Removing Cross Realm Trusts

Note: Adding realms to trust creates these entries in the krb5.conf file:

```
ITSO.IBM.COM = ??<
   ITSO.IBM.COM = THISREALM.COM
   ITSO.IBM.COM = THATREALM.COM
   ITSO.IBM.COM = EVERYREALM.COM
 ??>
```

5.3 Kerberos Client tasks through Qshell Interpreter

Kerberos related functions are available on the iSeries through the Qshell interpreter when the iSeries is acting as a client in a Kerberos enabled network. Each of the commands has a corresponding iSeries command line command that can be used interactively or inside your own programs. The sections that follow will list the Qshell and command line commands, a detailed description of each of the commands, the arguments available and authorities required to issue each of the commands.

Tip: To use Qshell commands you must be using the Qshell interface. This can be accomplished by entering the Start Qshell (STRQSH) command on the iSeries command line.

5.3.1 Using the kinit command

The kinit command is used to obtain or renew ticket granting tickets. When a new ticket is requested the credentials cache is re-initialized with the TGT received from the KDC. The Qshell and corresponding iSeries command, respectively, are:

- **`kinit`**
- **`call qsys/qkrbkinit parm('-l' ' 5h' 'principalName')`**

The default *PUBLIC authority to the kinit command is *USE. The following list provides the possible parameters that can be supplied when using the kinit command. The syntax of the command is `kinit [-r time] [-R] [-p] [-f] [-A] [-l time] [-c cache] [-k] [-t keytab] [principal]`

- `-r time` - The time interval for renewing a ticket. The ticket can no longer be renewed after the expiration of this interval. The renew time must be greater than the end time. If this option is not specified, the ticket is not renewable (a renewable ticket may still be generated if the requested ticket lifetime exceeds the maximum ticket lifetime).
- `-R` - An existing ticket is to be renewed. When you renew an existing ticket, you cannot specify any other ticket options.
- `-p` - The ticket can be a proxy. If you do not specify this option, the ticket cannot be a proxy.
- `-f` - The ticket can be forwarded. If you do not specify this option, the ticket cannot be forwarded.
- `-A` - The ticket will not contain a list of client addresses. If you do not specify this option, the ticket will contain the local host address list. When an initial ticket contains an address list, it can be used only from one of the addresses in the address list.
- `-l time` - The ticket end-time interval. After this interval expires, the ticket cannot be used unless it has been renewed. If you do not specify this option, the interval is set to 10 hours.
- `-c cache` - The name of the credentials cache that the kinit command will use. If you do not specify this option, the command uses the default credentials cache.

- `-k` - The key for the ticket principal is to be obtained from a key table. If you do not specify this option, the system prompts you to enter the password for the ticket principal.
- `-t keytab` - The key table name. If you do not specify this option but do specify the -k option, the system uses the default key table. The -t option implies the -k option.
- `principal` - The ticket principal. If you do not specify the principal on the command line, the system obtains the principal from the credentials cache.

When you specify options, such as `-c`, in your command argument you need to make sure that you have the proper authority to the files that will affected by the command. `Table 5-1` lists the authorities required for each of the options in the kinit command.

Table 5-1 Authorities to objects required when using the kinit command

Object referred to in command	**Authority Required**
Each directory in the path name preceding the key table file if -t option is specified	*X
Key table file when -t is specified	*R
Each directory in the path name preceding the credentials cache file to be used	*X
Parent directory of the cache file to be used, if specified by the KRB5CCNAME environment variable, and the file is being created (see note following table)	*WX
Credentials cache file	*RW
Each directory in the paths to the configuration files	*X
Configuration files	*R

Note: To enable the Kerberos run time to find your credentials cache file from any executing process, the name of the cache file is normally stored in the home directory in a file named krb5ccname. The storage location of the cache file name can be overridden by setting the environment variable _EUV_SEC_KRB5CCNAME_FILE. To access this file, the user profile must have *X authority to each directory in the path, and *R authority to the file where the cache file name is stored. The first time that a user creates a credentials cache, the user profile must have *WX authority to the parent directory.

5.3.2 Using the klist command

The klist command displays the contents of the credentials cache or keytab file. The Qshell command and corresponding command line command are:

- **`klist`**
- **`call qsys/qkrbklist parm('-a' '-e')`**

The default *PUBLIC authority to the klist command is *USE. The syntax for the klist command is `klist [-a] [-e] [-c] [-f] [-s] [-k] [-t] [-K] [filename]`.

- `-a` -Show all tickets in the credentials cache, including expired tickets. If you do not specify this option, expired tickets are not listed. This option is valid only when you list a credentials cache.
- `-e` - Display the encryption type for the session key and the ticket. This option is valid only when you list a credentials cache.
- `-c`- List the tickets in a credentials cache. If neither the -c nor the -k option is specified, this is the default. This option is mutually exclusive with the -k option.

- `-f` - Show the ticket flags, using the abbreviations listed in Table 5-2. This option is valid only when you list a credentials cache.

Table 5-2 Ticket Flag Abbreviations

Abbreviation	Meaning
F	Ticket can be forwarded
f	Forwarded ticket
P	Ticket can be a proxy
p	Proxy ticket
D	Ticket can be postdated
d	Postdated ticket
R	Renewable ticket
I	Initial ticket
i	Ticket not valid
A	Pre authentication used
O	Server can be a delegate
C	Transit list checked by the KDC

- `-s` - Suppress command output, but set the exit status to 0 if a valid ticket granting ticket is found in the credentials cache. This option is valid only when you list a credentials cache.
- `-k` - List the entries in a key table. This option is mutually exclusive with the -c option.
- `-t` - Display timestamps for key table entries. This option is valid only when you list a key table.
- `-K` - Display the encryption key value for each key table entry. This option is valid only when you list a key table.
- `filename` - Specifies the name of the credentials cache or key table. If no filename is specified, the default credentials cache or key table is used

The authorities required to access objects referred to in the options of the klist command are listed in Table 5-3.

Table 5-3 Authorities to objects required for the klist command

Object referred to in command	Authority Required
Each directory in the path name preceding the file if -k option is specified as keytab	*X
Keytab file when -k is specified	*R
Each directory in the path name preceding the credentials cache file if the -k is not specified	*X
Credentials cache file if the -k option is not specified	*R

5.3.3 Using the keytab command

The keytab command provides a method to add, list, and remove entries from the list of entries in the key table. The Qshell and command line commands for keytab are:

- **`keytab add krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM`**
- **`call qsys/qkrbkeytab parm('add' 'krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM')`**

If you do not specify the password with the `-p` option you will be prompted with `EUVF06048 Enter password:`. The default authority on the command is *PUBLIC *USE. The options for the add, list and remove commands are:

- `keytab add principal [-p password] [-v version] [-k keytab]`
- `keytab delete principal [-v version] [-k keytab]`
- `keytab list [principal] [-k keytab]`
  - `-k` - The key table name. If this option is not specified, the default key table is used.
  - `-p` - Specify the password. If this option is not specified, users are prompted to enter the password when they add an entry to the key table.
  - `-v` - The key version number. When you add a key, if this option is not specified, the next version number is assigned. When you delete a key, if this option is not specified, all keys for the principal are deleted.
  - `principal` - The principal name. When you list the key table, if this option is not specified, all principals are displayed.

Table 5-4 Authorities to objects required to use the keytab command

Object referred to in command	Authority required
Each directory in the path name preceding the target keytab file to be opened	*X
Parent directory of the target keytab file when add is specified, if the keytab file does not already exist	*WX
Keytab file when list is specified	*R
Target keytab file when add or delete is specified	*RW
Each directory in the paths to the configuration file	*X
Configuration files	*R

5.3.4 Using the kpasswd command

The kpasswd command is used to change the password for the specified Kerberos principal. The current password must be supplied in order to complete the process. Any specific password policies that have been configured on the KDC will be applied to the new password before the password is changed. The Qshell and command line commands are:

Note: You may not change the password for a ticket-granting service principal (krbtgt/realm) using the kpasswd command.

- **`kpasswd`**
- **`call qsys/qkrbkpsswd`**

After entering the command you are prompted for the current password. You are then required to enter the new password. The default authority for the kpasswd command is *PUBLIC *USE and the options are:

- `-A` - The initial ticket used by the kpasswd command will not contain a list of client addresses. The ticket will contain the local host address list if this option is not specified. When an initial ticket contains an address list, it can be used only from one of the addresses in the address list.
- `principal` - The principal whose password is to be changed. The principal will be obtained from the default credentials cache if the principal is not specified on the command line.

5.3.5 Using the kdestroy command

The kdestroy command deletes a Kerberos credentials cache file.The Qshell and command line commands for keytab are:

- **`kdestroy`**
- **`call qsys/qkrbkdstry parm('-c' 'myCache')`**

The default authority for the kdestroy command is PUBLIC *USE. The syntax of the command is `kdestroy [-c cache_name] [-e time_delta]`.

- `-c cache_name` - The name of the credentials cache to be destroyed. If no command options are specified, the default credentials cache is destroyed. This option is mutually exclusive with the -e option.
- `-e time_delta` - All credentials cache files that contain expired tickets are deleted if the tickets have been expired at least as long as the time_delta value.

Authorities

The default behavior is that the credentials cache file is created in the /QIBM/UserData/OS400/NetworkAuthentication/creds directory. The placement of the credentials cache file can be changed by setting the KRB5CCNAME environment variable.

If the credentials cache file has been created in another location use Table 5-5 to determine the authorities needed to each object and its data.

Table 5-5 Authorities required when credentials are not stored in the default location

Object referred to in command	Data Authority Required	Object Authority Required
Each directory in the path name preceding the credentials cache file	*X	None
Parent directory of the credentials cache file	*WX	None
Credentials cache file	*RW	*OBJEXIST
Each directory in the paths to the configuration files	*X	None
Configuration files	*R	None

Table 5-6 shows the authorities that are required when the credentials cache resides in the default location.

Table 5-6 Authorities required when credentials are stored in the default location

Object referred to in command	Data Authority Required	Object Authority Required
All directories in the path name	*X	None

Object referred to in command	Data Authority Required	Object Authority Required
Credentials cache file	*RW	None
Each directory in the path to the configuration files	*X	None
Configuration files	*R	None

5.3.6 Using the ksetup command

The ksetup command manages Kerberos service entries in the Directory Service (LDAP) directories. To use the ksetup command, you need to know the host name of the LDAP server, the LDAP administrator ID and the LDAP administrator's password. In this example the LDAP host name is AS20.ITSO.IBM.COM. The administrator is CN=ADMINISTRATOR and the password is LDAPPW.The Qshell and command line commands are:

- **`ksetup -h AS20.ITSO.IBM.COM -n CN=ADMINISTRATOR -p LDAPPW`**
- **`call qsys/qkrbksetup parm('-h' 'AS20.ITSO.IBM.COM' '-n' 'CN=ADMINISTRATOR' '-p' 'LDAPPW')`**

Once the KDC is successfully contacted you will get the following subcommand prompt:

```
ksetup>
```

The following subcommands can be run from the ksetup prompt:

- `addhost host-name realm-name` - This subcommand adds a host entry for the specified realm. The fully qualified host name should be used so that it resolves correctly no matter what default DNS domain is in effect on the Kerberos clients. If no realm name is specified, the default realm name is used.
- `addkdc host-name:port-number realm-name` - This subcommand adds a KDC entry for the specified realm. If a host entry does not already exist, one is created. If a port number is not specified, it is set to 88 . Use the fully qualified host name so that it resolves correctly no matter what default DNS domain is in effect on the Kerberos clients. If no realm name is specified, the default realm name is used.
- `delhost host-name realm-name` - This subcommand deletes a host entry and any associated KDC specification from the specified realm. If no realm name is specified, the default realm name is used.
- `delkdc host-name realm-name` - This subcommand deletes a KDC entry for the specified host. The host entry itself is not deleted. If no realm name is specified, the default realm name is used.
- `listhost realm-name` - This subcommand lists the host entries for a realm. If no realm name is specified, the default realm name is used.
- `listkdc realm-name` - This subcommand lists the KDC entries for a realm. If no realm name is specified, the default realm name is used.
- `exit` - This subcommand ends the ksetup command.

The default authority for the ksetup command is *PUBLIC *USE. The syntax of the command is `ksetup -h host-name -n bind-name -p bind-password -e` and the options are:

- `-h` - The host name for the Directory Service (LDAP) server. If you do not specify this option, the Directory Service (LDAP) server specified in the Kerberos configuration file is used.

- `-n` - The distinguished name to use when you bind to the Directory Service (LDAP) server. If you do not specify this option, the Directory Service (LDAP)_BINDDN environment variable is used to obtain the name.
- `-p` - The password to use when you bind to the Directory Service (LDAP) server. If this option is not specified, the Directory Service (LDAP)_BINDPW environment variable is used to obtain the password.
- `-e` - Echo each command line to stdout. This is useful when stdin is redirected to a file.

The authorities required to access objects referred to in the ksetup command are listed in Table 5-7.

Table 5-7 Authorities to objects required for the ksetup command

Object referred to in command	**Authority Required**
Each directory in the paths to the configuration files	*X
Configuration files	*R

5.4 More information

More information about Network Authentication Service on the iSeries can be found in the help text for iSeries Navigator and online at:

`http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm`

Select the appropriate **Geography -> Language (release) -> Security -> Network authentication service**.

Enterprise Identity Mapping

In 1.4, “SSO with Enterprise Identity Mapping” on page 9 we introduced Enterprise Identity Mapping (EIM) on a conceptual level. This chapter provides an overview of EIM, the components which make up the EIM domain, and how the interactions work between the EIM domain and its clients.

Enterprise Identity Mapping is still a fairly new concept to most people in the industry. EIM is a software solution that enhances single signon (SSO) across multiple platforms and multiple applications, while maintaining a secure environment. This open architecture allows administrators and application developers to solve the problems related to managing multiple IDs for users across the enterprise. What makes EIM so different from other SSO solutions is that no password synchronization or caching is involved in the EIM architecture.

EIM is an enabling technology infrastructure and not a solution by itself. As such, it becomes interesting to administrators primarily only when it is incorporated into a solution such as SSO. The infrastructure is provided on all IBM @server platforms today. The EIM architecture allows you to define relationships between individuals and entities in the enterprise and the various user IDs, employee numbers, telephone numbers, and so on. that represent those users. EIM offers a new approach to enable inexpensive solutions to a wide range of problems related to managing multiple user IDs for people and entities in the enterprise.

6.1 EIM overview

Today's network environments are made up of complex groups of systems and applications, resulting in the need to manage multiple user registries. Dealing with multiple user registries quickly grows into a large administrative problem that affects users, administrators, and application developers. Consequently, many companies are struggling to securely manage authentication and authorization for systems and applications. Enterprise Identity Mapping (EIM) is an IBM infrastructure technology that allows administrators and application developers to address this problem more easily and inexpensively than previously possible.

The following information describes the problems, outlines current industry approaches, and explains why solutions that exploit the EIM approach are better.

6.1.1 The problem of managing multiple user registries

Many administrators manage networks that include different systems and servers, each with a unique way of managing users through various user registries. In these complex networks, administrators are responsible for managing each user's identities and passwords across multiple systems. Additionally, administrators often must synchronize these identities and passwords, and users are burdened with remembering multiple identities and passwords and with keeping them in sync. The user and administrator overhead in this environment is excessive. Consequently, administrators often spend valuable time troubleshooting failed logon attempts and resetting forgotten passwords instead of managing the enterprise.

The problem of managing multiple user registries also affects application developers who want to provide multiple-tier or heterogeneous applications. These developers understand that customers have important business data spread across many different types of systems, with each system possessing its own user registries. Consequently, developers must create proprietary user registries and associated security semantics for their applications. Although this solves the problem for the application developer, it increases the overhead for users and administrators.

6.1.2 Current approaches

Several current industry approaches for solving the problem of managing multiple user registries are available, but they all provide incomplete solutions. For example, Lightweight Directory Access Protocol (LDAP) provides a distributed user registry solution. However, using LDAP (or other popular solutions such as Microsoft Passport) present administrators with the challenge of managing yet another user registry and the associated security semantics, or replacing existing applications that are built to use those registries.

Using this type of solution, administrators must manage multiple security mechanisms for individual resources, thereby increasing administrative overhead and potentially increasing the likelihood of security exposures. When multiple mechanisms support a single resource, the chances of changing the authority through one mechanism and forgetting to change the authority for one or more of the other mechanisms is much higher.

After completing this work, administrators find that they have not completely solved the problem. Generally, enterprises have invested too much money in current user registries and in their associated security semantics to make using this type of solution practical. Creating another user registry and associated security semantics solves the problem for the application provider, but not the problems for users or administrators.

One other possible solution is to use a single sign-on approach. Several products are available that allow administrators to manage files that contain all of a user's identities and passwords. However, this approach has several weaknesses:

- It addresses only one of the problems that users face. Although it allows users to sign on to multiple systems by supplying one identity and password, it does not eliminate the need for the user to have passwords on other systems, or the need to manage these passwords.
- It introduces a new problem by creating a security exposure because clear-text or decryptable passwords are stored in these files (passwords should never be stored in clear-text files or be easily accessible by anyone, including administrators).
- It does not solve the problems of third-party application developers that provide heterogeneous, multiple-tier applications. They must still provide proprietary user registries for their applications.

Despite these weaknesses, some enterprises have chosen to adopt these approaches because they provide some relief for the multiple user registry problems.

6.1.3 The EIM approach

EIM offers a new approach to enable inexpensive solutions to much more easily deal with multiple user registries and user identities in an enterprise. EIM is an architecture for describing the relationships between individuals or entities (like file servers and print servers) in an enterprise and the many identities that represent them. In addition, EIM provides a set of APIs that allow applications to ask questions about these relationships.

For example, given a person's user identity in one user registry, you can determine which user identity in another user registry represents that same person. If the user has authenticated with one user identity and you can map that user identity to the appropriate identity in another user registry, the user does not need to provide credentials for authentication again. You know who the user is and only need to know which user identity represents that user in another user registry. Therefore, EIM provides a generalized identity mapping function for the enterprise.

The ability to map between a user's identities in different user registries provides many benefits. Primarily, it means that applications may have the flexibility of using one user registry for authentication while using an entirely different user registry for authorization.

The use of identity mapping requires that administrators do the following:

- Create EIM identifiers that represent people or entities in their enterprise.
- Create EIM registry definitions that describe the existing user registries in their enterprise.
- Define the relationship between the user identities in those registries to the EIM identifiers that they created.

No code changes are required to existing user registries. The administrator does not need to have mappings for all identities in a user registry. EIM allows one-to-many mappings (a single user with more than one user identity in a single user registry). EIM also allows many-to-one mappings (multiple users sharing a single user identity in a single user registry, which although supported is not advised). An administrator can represent any user registry of any type in EIM.

EIM is an open architecture that administrators may use to represent identity mapping relationships for any registry. It does not require copying existing data to a new repository and trying to keep both copies synchronized. The only new data that EIM introduces is the relationship information. Administrators manage this data in an LDAP directory, which provides the flexibility of managing the data in one place and having replicas wherever the information is used. Ultimately, EIM gives enterprises and application developers the flexibility to easily work in a wider range of environments with less cost than would be possible without this support.

6.2 Benefits of single signon

In this section we discuss some of the benefits associated with SSO solutions that also exploit EIM. There are three primary areas that we address, users, administrators, and developers.

6.2.1 Benefits for users

In a single sign-on environment, authentication happens whenever users attempt to access a new system; however, they will not be prompted for IDs or passwords. An SSO solution that integrates EIM reduces the need for users to keep track of and manage multiple user names and passwords to access other systems in the network. Once a user is authenticated to the network, the user can access services and applications across the enterprise without the need for multiple passwords to these different systems.

6.2.2 Benefits for administrators

For an administrator, single sign-on simplifies overall security management of an enterprise. Without single sign-on, users and applications may cache passwords to different systems, which can compromise the security of the entire network. Administrators spend time and money on solutions to diminish these security risks. Single sign-on reduces the administrative overhead in managing authentication while continuing to employ the access control mechanism you already have in place; therefore keeping your entire network every bit as secure as it was before implementing SSO. Additionally, single sign-on reduces the administrative costs of resetting forgotten passwords.

6.2.3 Benefits for application developers

For developers of applications that must run in heterogeneous networks, EIM provides the infrastructure to develop applications that work across platforms. Application developers can, for reduced costs, build products that support a wider range of authentication mechanisms without having to build any application specific authorization mechanisms. This, in turn, produces an application which is capable of being integrated into a larger number of customer environments with less overhead. In other words, by incorporating EIM, application developers can more cost effectively implement multi-tiered, cross-platform applications that are cheaper for a wider range of customers to deploy.

6.3 EIM components

There are a number of components that comprise EIM as we have seen from the overview above. Figure 6-1 on page 75 shows how the different components described above fit together to provide a single signon environment.

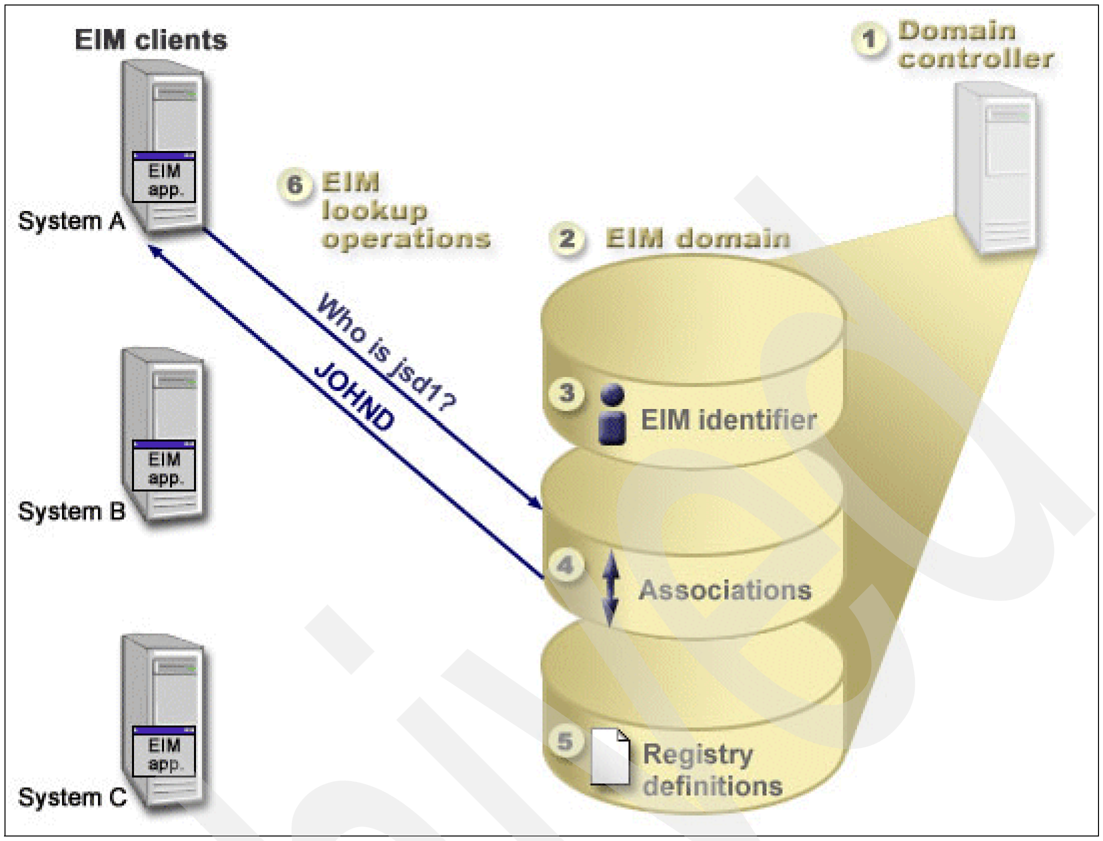


Figure 6-1 An overview of the components of EIM

In the coming sections within this chapter, we explain, in depth, what the different components in this EIM hierarchy represent and how they are used within the EIM infrastructure.

6.3.1 EIM domain controller

At the top level of the EIM architecture is the domain controller (item 1 in Figure 6-1). The EIM domain controller is a Lightweight Directory Access Protocol (LDAP) server which is configured to manage at least one EIM domain. The domain controller is implemented as a tree structure in the LDAP server. You can only have one domain controller per physical server or logical partition (LPAR), provided your server supports the LPAR capability. To enable EIM in your enterprise, a minimum of one domain controller must exist. If you have multiple EIM domains in your enterprise you can use one domain controller to link all of the EIM domains created on different servers.

Any of the IBM @server platforms can act as a domain controller, provided they are running the IBM Directory Server. The critical thing for a server being able to act as a domain controller is that the system has proper LDAP support. Without this support, a server cannot be used as a domain controller. Below are some examples of servers that can provide the EIM domain controller because of their LDAP support:

- iSeries, formerly known as the AS/400
- zSeries, formerly known as the S/390®
- pSeries®, formerly known as RS/6000®
- xSeries®, also known as Windows
- Linux
- Open LDAP

6.3.2 EIM domain

The domain, as shown as item 2 in Figure 6-1, contains EIM identifiers, registries for each of the different user registries in an organization, and mapped associations between specific user IDs in the user registries and the EIM identifier they are associated with.

As EIM is implemented in a tree structure there is a node or branch in this tree which represents an individual EIM domain. The EIM domain can either be connected to a parent node or it can connected directly to the root of the LDAP tree. Most companies will only require one EIM domain to host their EIM implementation. The EIM domain is created through the iSeries Navigator interface. Section 7.2.2, “Add the EIM domain to be managed” on page 116 provides details on configuring an EIM domain.

An EIM domain is different from a user registry. A user registry defines a set of user identities known to and trusted by a particular instance of an operating system or application. A user registry also contains the information needed to authenticate the user of the identity. Additionally, a user registry often contains other attributes such as user preferences, system privileges, or personal information for that identity. In contrast, an EIM domain *refers* to user identities that are defined in user registries. An EIM domain contains information about the *relationship* between identities in various user registries (user name, registry type, and registry instance) and the actual people or entities that these identities represent. Because EIM tracks relationship information only, there is nothing to synchronize between user registries and EIM. This segregation of passwords and information one major benefit of EIM. Passwords can be changed in specific user registries on a system and nothing needs to be changed within EIM.

6.3.3 EIM identifiers

Each unique entity that is added to an EIM domain is represented as an EIM identifier (item 3 in Figure 6-1 on page 75). An EIM identifier can represent a person, a server, a printer, or a number of other resources. An EIM identifier and associations are required before that entry can use the EIM infrastructure. Figure 6-2 provides an example of a person in the organization named Kim Greene. Kim will need to be added to the EIM domain as an EIM identifier. Once this identifier is added, associations can then be created mapping which user IDs Kim Greene has on the various servers throughout the organization.

Figure 6-2 shows that EIM identifier Kim Greene has user identities in different registries:

- kimgreene
- kkg1
- KIMG
- KGreene

Associations need to be made to map these various user identifies to the Kim Greene EIM identifier. By creating these associations, EIM lookup operations can be performed to retrieve the correct user identity for Kim Greene based on which system is being accessed.

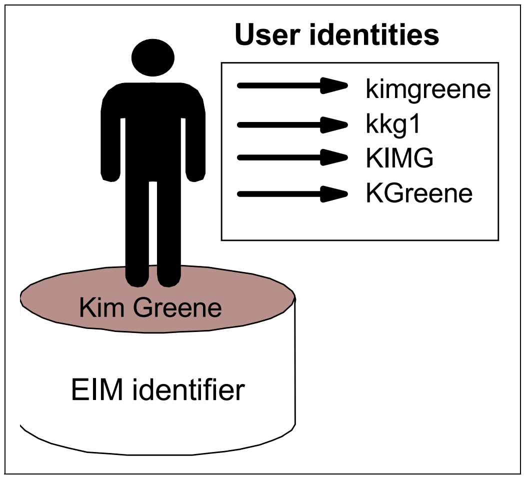


Figure 6-2 EIM identifier

An EIM identifier needs to be unique within a domain. The identifier does not have to be completely comprehensive, but we recommend that each EIM identifier is called something unique to the specific entry type such as an employee number, fully qualified domain name, or TCP/IP address.

EIM identifier representing an entity

In addition to representing users, EIM identifiers can represent entities within your enterprise as Figure 6-3 illustrates. For example, often the print server function in an enterprise runs on multiple systems. In Figure 6-3, the print server function in the enterprise runs on three different systems under three different user identities of pserverID1, pserverID2, and pserverID3.

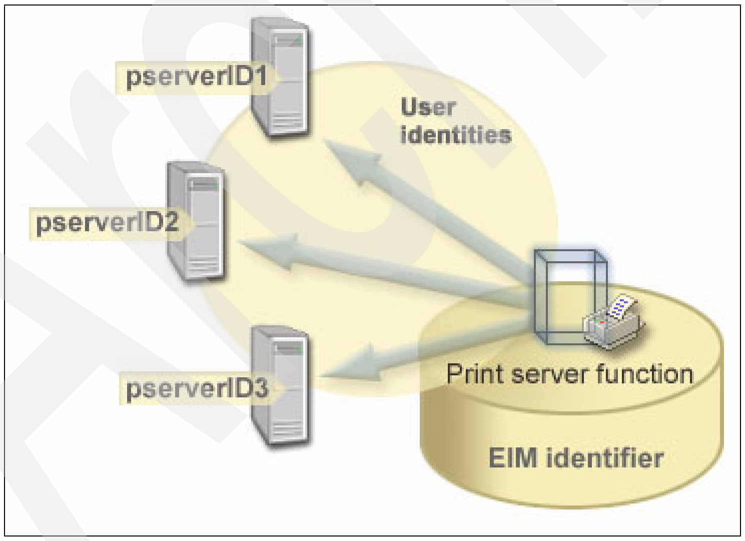


Figure 6-3 The relationship between the EIM identifier that represents the print server function and the various user identities for that function

With EIM, you can create a single identifier that represents the print server function within the entire enterprise. In this example, the EIM identifier print server function represents the actual print server function entity in the enterprise. Associations are created to define the relationships between the EIM identifier (print server function) and each of the user identities for this function (pserverID1, pserverID2, and pserverID3). These associations allow application developers to use EIM lookup operations to find a specific print server function. Application providers can then write distributed applications that manage the print server function more easily across the enterprise.

EIM identifiers and aliasing

You can also create aliases for EIM identifiers. Aliases can aid in locating a specific EIM identifier when performing an EIM lookup operation. One use of an alias may be a situation where an individual's legal name is different from the name that person is known as. An alias can be created to allow that individual to be referred to by their more commonly known name.

EIM identifier names must be unique within an EIM domain. Aliases can help address situations where using unique identifier names can be difficult. For example, different individuals within an enterprise can share the same name, which can be confusing if you are using proper names as EIM identifiers. Figure 6-4 illustrates an example in which an enterprise has two users named John S. Day. The EIM administrator creates two different EIM identifiers to distinguish between them: John S. Day1 and John S. Day2. However, which John S. Day is represented by each of these identifiers is not readily apparent.

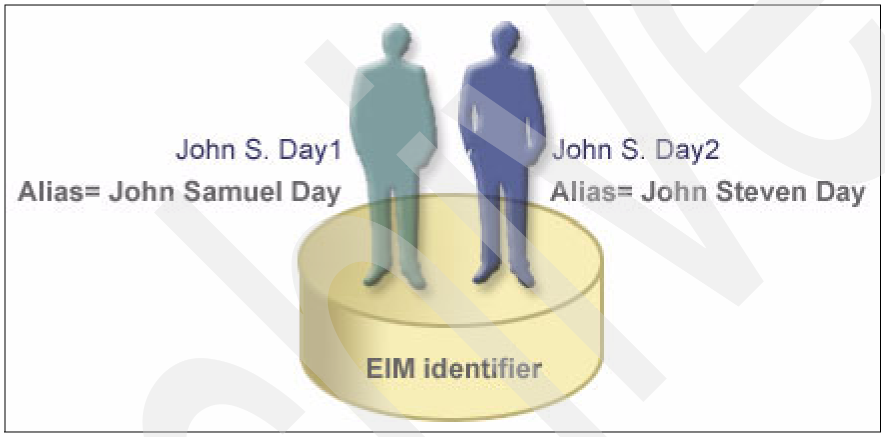


Figure 6-4 Aliases for two EIM identifiers based on the shared proper name John S. Day

By using aliases, the EIM administrator can provide additional information about the individual for each EIM identifier. This information can also be used in an EIM lookup operation to distinguish between the users that the identifier represents. For example, the alias for John S. Day1 might be John Samuel Day and the alias for John S. Day2 might be John Steven Day.

Each EIM identifier can have multiple aliases to identify which John S. Day the EIM identifier represents. The EIM administrator might add another alias to each of the EIM identifiers for the two individuals to further distinguish between them. For example, the additional aliases might contain each user's employee number, department number, job title, or some other distinguishing attribute.

6.3.4 EIM registry definitions

An EIM registry definition (item 5, Figure 6-1 on page 75) represents an actual user registry that exists on a system within the enterprise. A user registry operates like a directory and contains a list of valid user identities for a particular system or application. A basic user registry contains user identities and their passwords. One example of a user registry is the z/OS Security Server Resource Access Control Facility (RACF®) registry. User registries can contain other information as well. For example, a Lightweight Directory Access Protocol (LDAP) directory contains bind distinguished names, passwords, and access controls to data that is stored in LDAP. Other examples of common user registries are a Kerberos key distribution center (KDC) and the OS/400 user profiles registry.

EIM registry definitions provide information regarding those user registries in an enterprise. The administrator defines these registries to EIM by providing the following information:

- A unique, arbitrary EIM registry name
- The type of user registry

Each registry definition represents a specific instance of a user registry. Consequently, you should choose an EIM registry definition name that helps you to identify the particular instance of the user registry. For example, you could choose the TCP/IP host name for a system user registry, or the host name combined with the name of the application for an application user registry. You can use any combination of alphanumeric characters, mixed case, and spaces to create unique EIM registry definition names.

In Figure 6-5, the administrator created EIM registry definitions for user registries representing System A, System B, and System C. System A contains a user registry for WebSphere Lightweight Third-Party Authentication (LTPA). The registry definition name that the administrator uses helps to identify the specific occurrence of the type of user registry. An IP address or host name is often sufficient for many types of user registries. In this example, the administrator identifies the specific user registry instance by using System_A_WAS as the registry definition name. In addition to the name, the administrator also provides the type of registry as WebSphere LTPA.

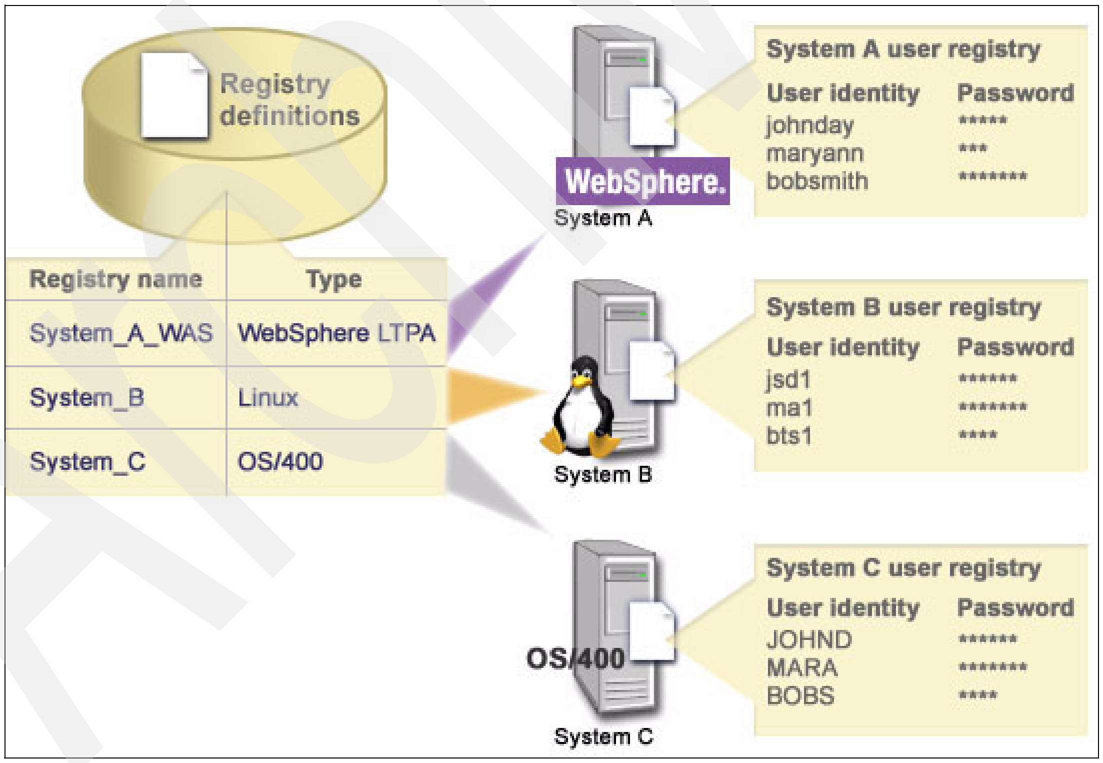


Figure 6-5 EIM registry definitions for three user registries in an enterprise

You can also define user registries that exist within other user registries. For example, the z/OS Security Server (RACF) registry can contain specific user registries that are a subset of users within the overall RACF user registry. There are two types of registry that we will discuss now. These are application registries and system registries.

System and application registry definitions

Some applications use a subset of user identities within a single instance of a user registry. EIM allows administrators to model this scenario by providing two kinds of EIM registry definitions: system and application.

A system registry definition represents a distinct registry within a workstation or server. You can create a system registry definition when the registry in the enterprise has one of the following traits:

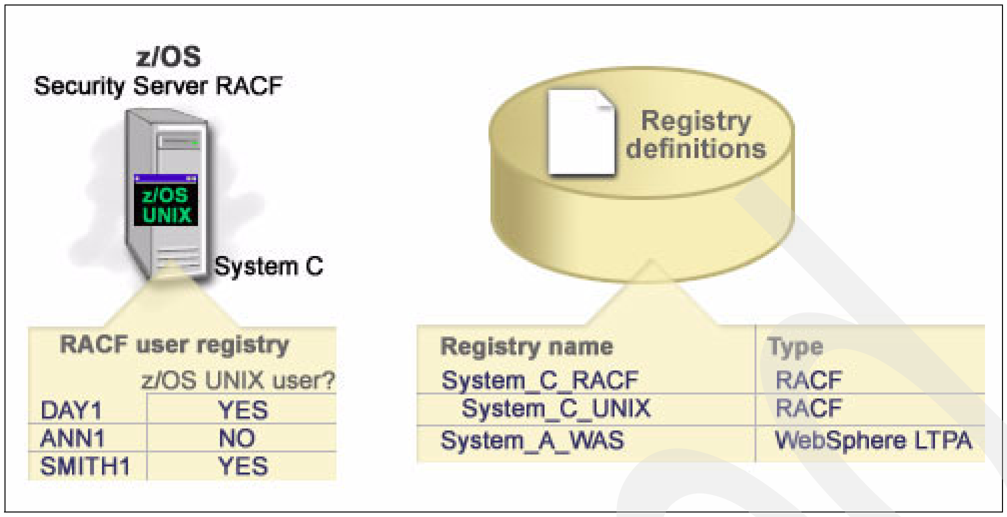
- The registry is provided by an operating system, such as AIX, OS/400, or a security management product such as z/OS Security Server Resource Access Control Facility (RACF).
- The registry contains user identities that are unique to a specific application, such as Lotus Notes.
- The registry contains distributed user identities, such as Kerberos principals or Lightweight Directory Access Protocol (LDAP) distinguished names.

An application registry definition represents a subset of user identities that are defined in a system registry. These user identities share a common set of attributes or characteristics that allow them to use a particular application or set of applications. You can create an application registry definition when the user identities have the following traits:

- The user identities for the application or set of applications are not stored in a user registry specific to the application or set of applications.
- The user identities for the application or set of applications are stored in a system registry that contains user identities for other applications.

EIM lookup operations perform correctly regardless of whether an EIM administrator defines a registry either as system or application. However, separate registry definitions allow mapping data to be managed on an application basis. The responsibility of managing application-specific mappings can be assigned to an administrator for a specific registry.

For example, Figure 6-6 shows how an EIM administrator created a system registry definition to represent a z/OS Security Server RACF registry. The administrator also created an application registry definition to represent the user identities within the RACF registry that use z/OS UNIX System Services (z/OS UNIX). System C contains a RACF user registry that contains information for three user identities, DAY1, ANN1, and SMITH1. Two of these user identities (DAY1 and SMITH1) access z/OS UNIX on System C. These user identities are actually RACF users with unique attributes that identify them as z/OS UNIX users. Within the EIM registry definitions, the EIM administrator defined System_C_RACF to represent the overall RACF user registry. The administrator also defined System_C_UNIX to represent the user identities that have z/OS UNIX attributes.



RACF user registry	z/OS UNIX user?
DAY1	YES
ANN1	NO
SMITH1	YES

Registry name	Type
System_C_RACF	RACF
System_C_UNIX	RACF
System_A_WAS	WebSphere LTPA

Figure 6-6 EIM registry definitions for the RACF user registry and for users of z/OS UNIX

EIM registry definitions and aliasing

You can also create aliases for EIM registry definitions. You can use predefined alias types or you can define your own alias types to use. The predefined alias types include:

- Domain Name System (DNS) host name
- Kerberos realm
- Issuer distinguished name (DN)
- Root distinguished name (DN)
- TCP/IP address
- LDAP DNS host name

This alias support allows programmers to write applications without having to know in advance the arbitrary EIM registry name chosen by the administrator who deploys the application. Application documentation can provide the EIM administrator with the alias name that the application uses. Using this information, the EIM administrator can assign this alias name to the EIM registry definition that represents the actual user registry that the administrator wants the application to use.

When the administrator adds the alias to the EIM registry definition, the application can perform an alias lookup to find the EIM registry name at initialization. The alias lookup allows the application to determine the EIM registry name or names to use as input to the APIs that perform EIM lookup operations.

6.3.5 EIM associations

An EIM association is a relationship between an EIM identifier that represents a specific person and a single user identity in a user registry that also represents that person. When you create associations between an EIM identifier and all of a person's or entity's user identities, you provide a single, complete understanding of how that person or entity uses the resources in an enterprise. EIM provides APIs that allow applications to find an unknown user identity in a specific (target) user registry by providing a known user identity in some other (source) user registry. This process is called identity mapping.

Before you can create an association, you first must create the appropriate EIM identifier and the appropriate EIM registry definition for the user registry that contains the associated user identity. Section 7.2.3, "Using iSeries Navigator to add identifiers and associations" on page 117 demonstrates how to create identifiers and associations. An association defines a relationship between an EIM identifier and a user identity by using the following information:

- EIM identifier name
- User identity name
- EIM registry definition name
- Association type

An administrator can create different types of associations between an EIM identifier and a user identity based on how the user identity is used. User identities can be used for authentication, authorization, or both.

Authentication is the process of verifying that an entity or person who provides a user identity has the right to assume that identity. Verification is often accomplished by forcing the person who submits the user identity to provide secret or private information associated with the user identity, such as a password.

Authorization is the process of ensuring that a properly authenticated user identity can only perform functions or access resources for which the identity has been given privileges. In the past, nearly all applications were forced to use the user identities in a single user registry for both authentication and authorization. By using EIM lookup operations, applications now can use user identities in one user registry for authentication while using associated user identities in a different user registry for authorization.

In EIM, there are three types of associations that an administrator can define between an EIM identifier and a user identity. These types are source, target, and administrative associations.

Source association

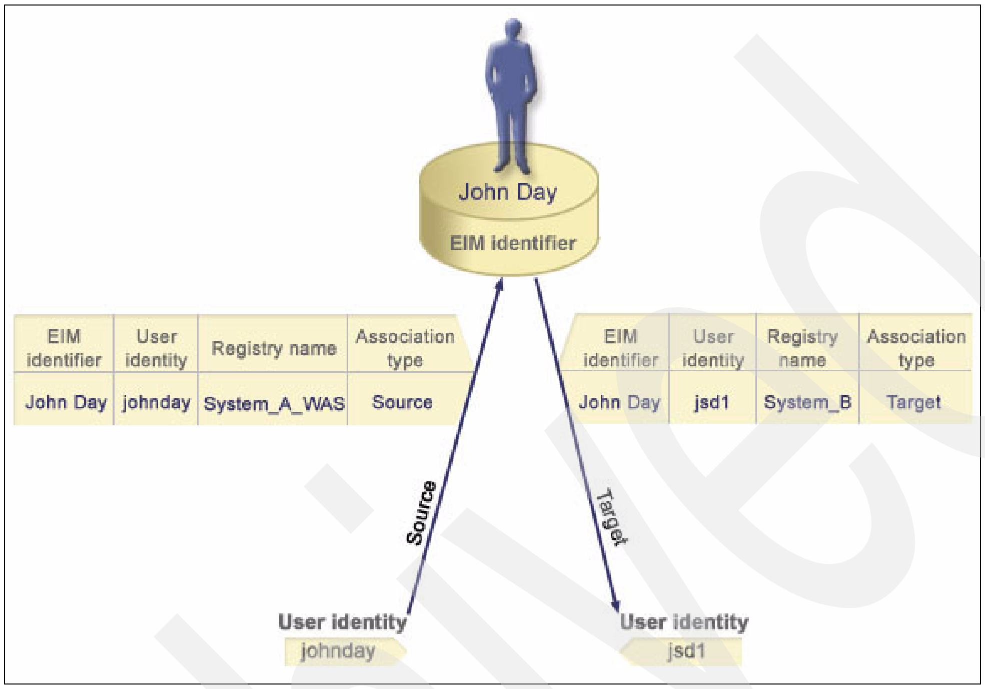
When a user identity is used for authentication, such as your Windows login, that user identity should have an EIM Identifier that uses that authenticated identity as a source association. A source association allows the user identity to be used as the source in an EIM lookup operation to find a different user identity (target) that is associated with the same EIM Identifier. In order for an EIM Identifier to have any benefit there must be at least one source association and one target association.

Target association

When a user identity is used for authorization rather than for authentication, that user identity should have an EIM Identifier that uses that ID as a target association. A target association allows the user identity to be returned as the result of an EIM lookup operation. If a user identity with only a target association is used as the source identity in an EIM lookup operation, no associated user identities are returned.

It may be necessary to create both a target and a source association for a single user identity. This is required when an individual uses a single system as both a client and a server or for individuals who act as administrators. For example, a user normally authenticates to a Windows platform and runs applications that access an AIX server. Because of the user's job responsibilities, the user must occasionally also log directly into an AIX server. In this situation you would create both source and target associations between the AIX user identity and the person's EIM identifier. User identities that represent end users normally need a target association only.

Figure 6-7 shows an example of a source and a target association. In this example, the administrator created two associations for the EIM identifier John Day to define the relationship between this identifier and two associated user identities. The administrator created a source association for johnday, the WebSphere Lightweight Third-Party Authentication (LTPA) user identity in the System_A_WAS user registry. The administrator also created a target association for jsd1, the OS/400 user profile in the System B user registry. These associations provide a means for applications to obtain an unknown user identity (the target, jsd1) based on a known user identity (the source, johnday) as part of an EIM lookup operation.



EIM identifier	User identity	Registry name	Association type
John Day	johnday	System_A_WAS	Source

EIM identifier	User identity	Registry name	Association type
John Day	jsd1	System_B	Target

Figure 6-7 EIM target and source associations for the EIM identifier John Day

Administrative association

An administrative association for an EIM identifier is typically used to show that the person or entity represented by the EIM identifier owns a user identity that requires special considerations for a specified system. This type of association can be used, for example, with highly sensitive user registries.

Due to the nature of what an administrative association represents, an EIM lookup operation that supplies a source user identity with an administrative association returns no results. Similarly, a user identity with an administrative association is never returned as the result of an EIM lookup operation. It could be a serious issue if an administrative association is returned in the case of root on linux or QSECOFR on OS400.

Figure 6-8 shows an example of an administrative association. In this example, John Day has one user identity on System A and another user identity on System B, which is a highly secure system. The system administrator wants to ensure that users authenticate to System B by using only the local user registry of this system. The administrator does not want to allow an application to authenticate John Day to the system by using some foreign authentication mechanism. By using an administrative association for the JDay user identity on System B, the EIM administrator can see that John Day owns an account on System B, but EIM does not return information about the JDay identity in EIM lookup operations. Even if applications exist on this system that use EIM lookup operations, they cannot find user identities that have administrative associations.

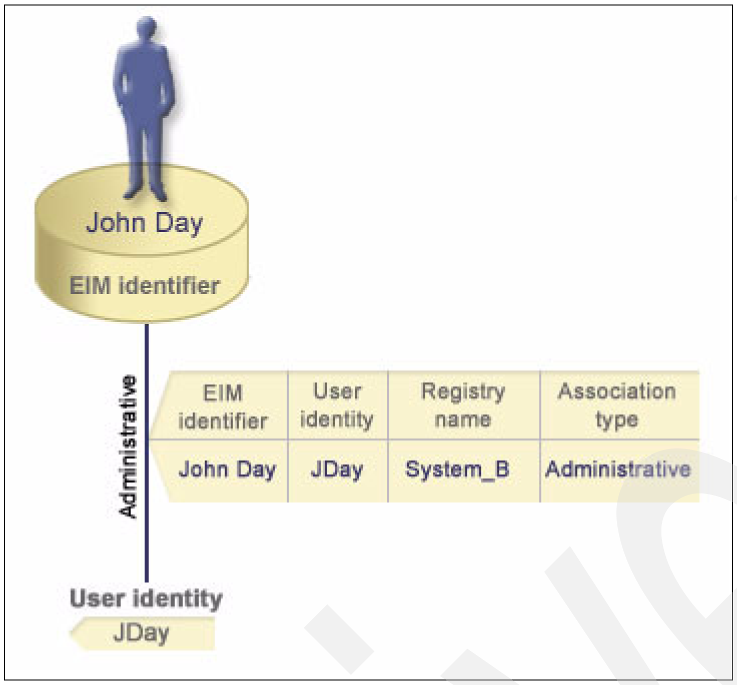


Figure 6-8 EIM administrative association for the EIM identifier John Day

6.3.6 EIM lookup operations

An EIM lookup operation is a process through which an application or operating system finds an unknown associated user identity in a specific target registry by supplying some known and trusted information. Applications that use EIM APIs can perform these EIM lookup operations on information only if that information is stored in the EIM domain. An application can perform one of two types of EIM lookup operations based on the type of information the application supplies as the source of the EIM lookup operation: a user identity or an EIM identifier.

When an application supplies a user identity as the source, the application also must supply the EIM registry definition name for the source user identity and the EIM registry definition name that is the target of the EIM lookup operation. To be used as the source in a EIM lookup operation, a user identity must have a source association defined for it.

When an application supplies an EIM identifier as the source of the EIM lookup operation, the application must also supply the EIM registry definition name that is the target of the EIM lookup operation. For a user identity to be returned as the target of either type of EIM lookup operation, the user identity must have a target association defined for it.

The supplied information is passed to the EIM domain controller where all EIM information is stored and the EIM lookup operation searches for the source association that matches the supplied information. Based on the EIM identifier (supplied to the API or determined from the source association information), the EIM lookup operation then searches for a target association for that identifier that matches the target EIM registry definition name.

In Figure 6-9, the user identity johnday authenticates to the WebSphere Application Server by using Lightweight Third-Party Authentication (LPTA) on System A. The WebSphere Application Server on System A calls a native program on System B to access data on System B. The native program uses an EIM API to perform an EIM lookup operation based on the user identity on System A as the source of the operation. The application supplies the following information to perform the operation: johnday as the source user identity, System_A_WAS as the source EIM registry definition name, and System_B as the target EIM registry definition name. This source information is passed to the EIM domain controller and the EIM lookup operation finds a source association that matches the information. Using the EIM identifier name, the EIM lookup operation searches for a target association for the John Day identifier that matches the target EIM registry definition name for System_B. When the matching target association is found, the EIM lookup operation returns the jsd1 user identity to the application.

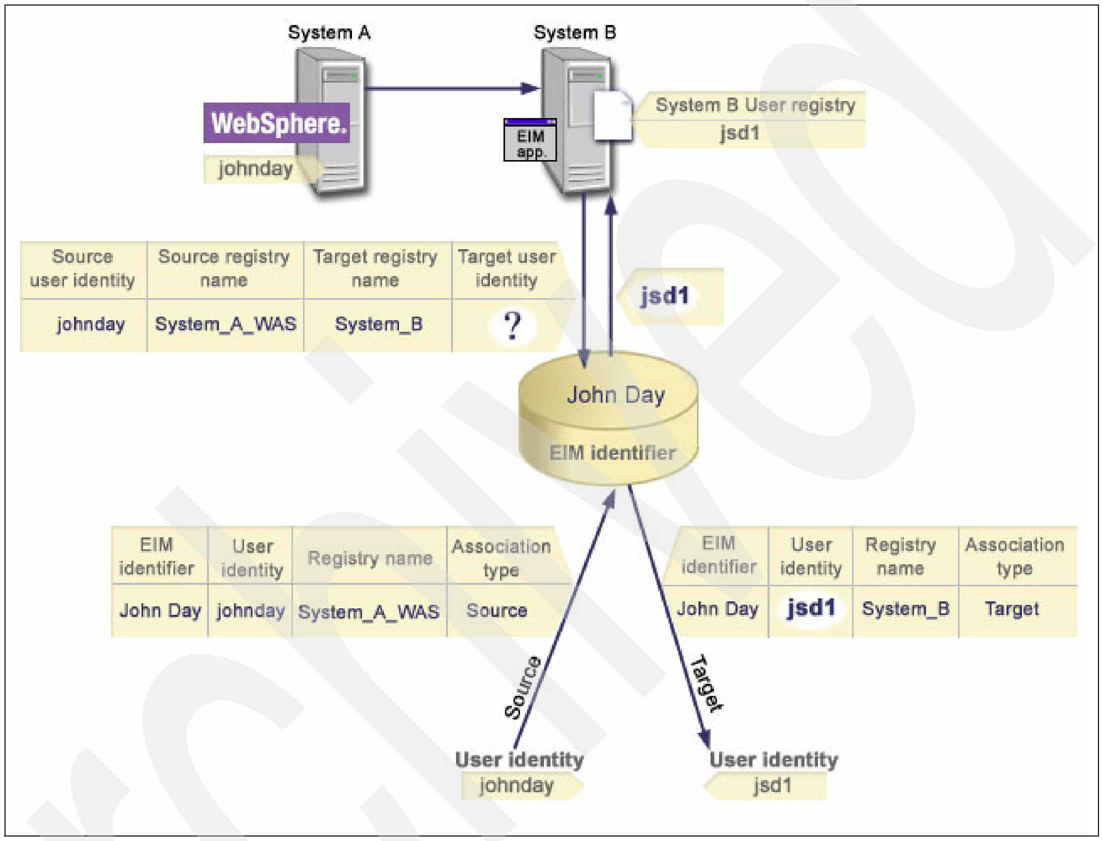


Figure 6-9 EIM lookup operation based on the known user identity johnday

An application that performs a search such as the one shown in Figure 6-9 can find out what a specific user's ID is on another system. This EIM lookup operations enables an application requiring SSO capabilities to access resources on another system. By using EIM, the user is not required to provide their password for access to iSeries System B. Additionally, the application that is running on behalf of the user can only access resources that the target user identity has authority to access on iSeries System B.

6.3.7 EIM authorities

There are different authority groups within EIM that allow you to perform different functions, depending on which group is being used to access and work with the EIM environment. The EIM administrator groups include:

- Lightweight directory access protocol (LDAP) administrator
- EIM administrator
- EIM identifiers administrator
- EIM mapping lookup
- EIM registries administrator

- EIM registry X administrator

Let’s look at each of these administrator groups to see what they are allowed to do within the EIM architecture. The LDAP administrator group has the highest level of authority. You will see as we progress down the list, the level of authority for each administrator group diminishes.

LDAP administrator

A user with LDAP administrator authority can perform the following operations:

- Create a domain
- Delete a domain
- Create and remove EIM identifiers
- Create and remove an EIM registry definition
- Create and remove source, target, and administrative associations
- Perform EIM lookup operations
- Retrieve associations, EIM identifiers, and EIM registry definitions from the EIM domain
- Add, remove, and list EIM authority information

EIM administrator

The EIM administrator authority level allows a user to manage an EIM domain and all of the data contained within the domain. A user with this authority can perform these functions:

- Delete a domain
- Create and remove EIM identifiers
- Create and remove an EIM registry definition
- Create and remove source, target, and administrative associations
- Perform EIM lookup operations
- Retrieve associations, EIM identifiers, and EIM registry definitions
- Add, remove, and list EIM authority information

EIM identifiers administrator

This level of authority allows a user to manage EIM identifiers and associations mapped to those identifiers. A user with this level of authority can perform the following operations:

- Create an EIM identifier
- Add and remove source associations
- Add and remove administrative associations
- Retrieve associations, EIM identifiers, and EIM registry definitions

EIM mapping lookup

A user with this level of authority can perform the following lookup operations:

- Perform EIM lookup operations
- Retrieve associations, EIM identifiers, and EIM registry definitions

EIM registries administrator

This level of authority allows a user to perform EIM registry definition management types of functions including:

- Adding and removing target associations
- Performing EIM lookup operations
- Retrieving associations, EIM identifiers, and EIM registry definitions

EIM registry X administrator

A user with this level of authority can manage a specific EIM registry definition. An EIM registry X administrator can perform the following operations:

- Add and remove target associations for the EIM registry definition
- Perform EIM lookup operations
- Retrieve associations, EIM identifiers, and EIM registry definitions

6.3.8 Setting Up EIM Authorities

As discussed in the previous section, EIM authorities are access groups which define the operations that members of those groups can perform. These access groups are not limited just to users. Below you will see that entities can be given an authority based on three different selection criteria.

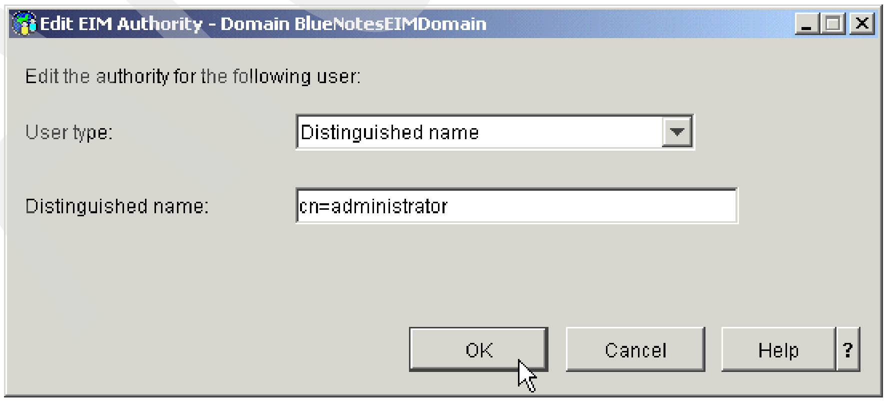
This is the first step in configuring access for an authority. The administrator should navigate to the relevant EIM domain though the steps: Your **iSeries -> Networks -> Enterprise Identity Mapping -> Domain Management** then select your EIM Domain and right-click on it. Select **Authority** from the local menu.

After the Authority item is clicked, a new window will open asking you to select the entity that the EIM authority would like to be either created or modified for. There are three different ways that users can be selected from this window:

- Distinguished name
- Kerberos principal name
- OS400 user profile

These three options allow for the assigning of authorities to users and also services and servers. Despite the way that an entity is selected in this list it has a distinguished name equivalent associated with it. This could be used in the distinguished name field rather than changing the type to Kerberos or OS400 user profile. The advantages that changing these values other than the distinguished name field is the quick access to an entities properties. For an example of these distinguished names returned from Kerberos and OS400 user profiles, see 9.5.3, “Using the UserAccess class” on page 218.

Figure 6-10 is the dialog box as presented when selecting an entity through a distinguished name.


Edit EIM Authority - Domain BlueNotesEIMDomain

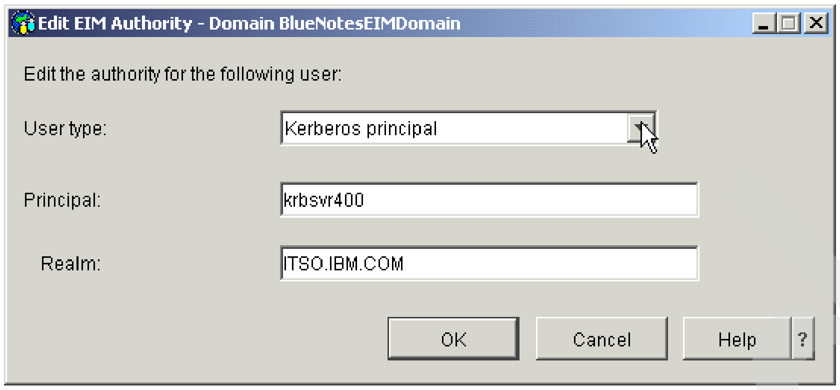
Edit the authority for the following user:

User type: Distinguished name

Distinguished name: cn=administrator

Figure 6-10 Select entity by distinguished name

Figure 6-11 is the dialog box as presented when selecting an entity through a Kerberos principal name.

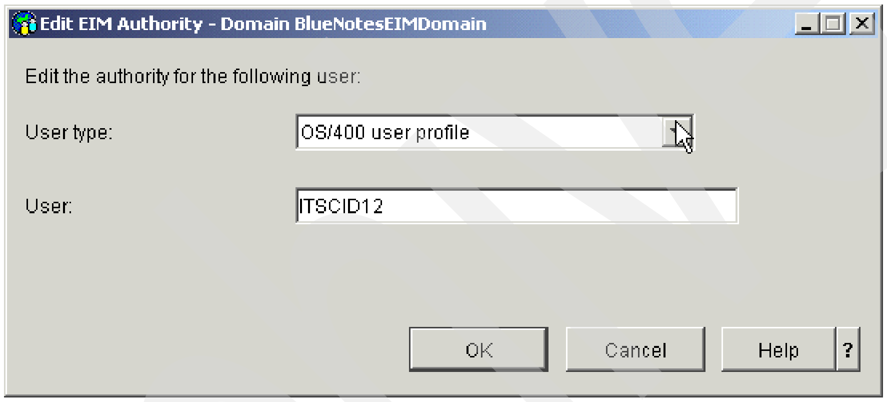

Edit EIM Authority - Domain BlueNotesEIMDomain

Edit the authority for the following user:

User type:	Kerberos principal
Principal:	krbsvr400
Realm:	ITSO.IBM.COM

Figure 6-11 *Selecting an entity by a Kerberos principal name*

Figure 6-12 is the dialog box as presented when selecting an entity through an OS400 user profile.


Edit EIM Authority - Domain BlueNotesEIMDomain

Edit the authority for the following user:

User type:	OS/400 user profile
User:	ITSCID12

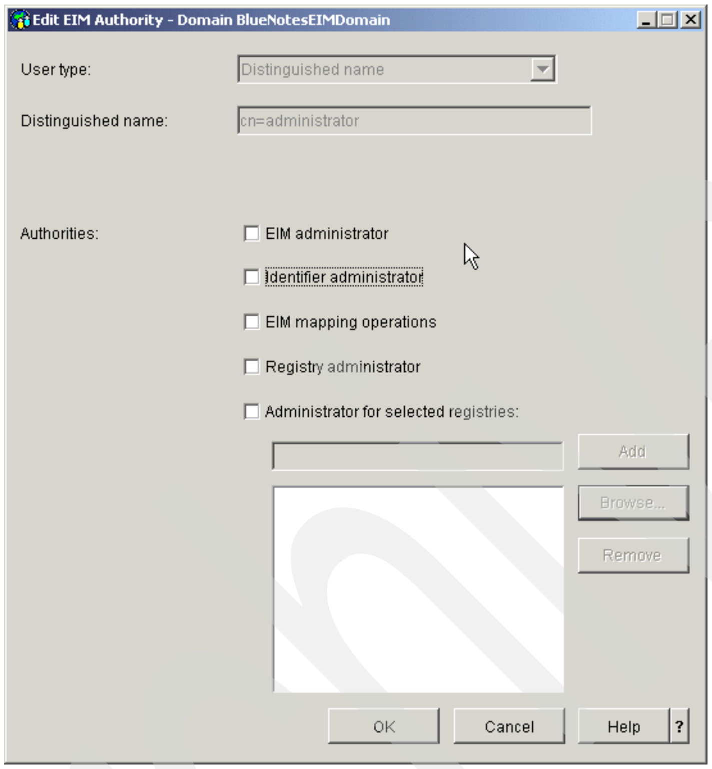
Figure 6-12 *Selecting an entity by a OS400 user profile*

After an entity has been selected the properties window for that user is then displayed (Figure 6-13). A list of authorities and the functions that they can perform can be found in the previous section, 6.3.7, “EIM authorities” on page 85. The level of access for the selected entity is controlled through a list of check boxes. If the EIM Administrator check box is selected then all other check boxes are grayed out. If this authority is selected then the entity will have access to perform all the other functions associated with other users, allowing other check boxes to be selected could be troublesome.

The Identity Administrator, EIM mapping operations and Registry administrator check boxes can be selected independently allowing for the combination of different authorities to be merged together. This has the advantage that there does not need to be separate entities for all different authorities, but allows the flexibility to have the option of doing this. This can be very advantageous depending on the size of an organization or based on the security and audit procedures imposed by an organization.

If the Registry administrator check box is selected then the Administrator for selected registries box and buttons to add registries are grayed out. This is because the registry administrator authority has access to create, change and delete registries.

If the Registry administrator box is not selected then the option to become the administrator for selected registries check box is enabled. When this check box is selected the list functions at the bottom of this screen are enabled. Registries can be added from typing in the name of the registry in the text box and clicking add. Registries can also be removed from this list by selecting the value in this list and clicking remove. When the browse button is clicked, a list of available registries in EIM is displayed. This is similar to the browse registries button which is available when adding an association for an identifier. This allows for multiple registries to be selected and when the OK button is clicked, these selected registries appear in the list box on the properties window. There is also a help button at the bottom of the window to bring up a dialog window. The other option is to click the question mark at the bottom of the window and this will give you specific information about the item clicked.


Edit EIM Authority - Domain BlueNotesEIMDomain

User type: Distinguished name

Distinguished name: cn=administrator

Authorities:
- [ ] EIM administrator
- [ ] Identifier administrator
- [ ] EIM mapping operations
- [ ] Registry administrator
- [ ] Administrator for selected registries:

Figure 6-13 EIM authorities properties for an entity

The OK and Cancel buttons can be clicked to either confirm the selections made or to cancel what has been done on this properties window.

6.4 APIs available to work with the EIM environment

Aside from the currently EIM enabled applications listed here you will also want to incorporate and develop EIM functionality into your own applications.

- iSeries Navigator
- DRDA
- PC5250 and Telnet
- NetServer
- QFileSvr.400

If you are working with an application that is not already single signon enabled, you can do so by using specific APIs. The APIs that exist for these development tasks are the C and Java APIs. More information can be found about these APIs in Chapter 9, "Programming APIs and examples" on page 187.

6.5 Three steps to success

To make EIM usable, there are potentially three steps which could be followed to add information into EIM:

- Collection
- Collation
- Population

Figure 6-14 shows a pictorial representation of the three processes outlined above.

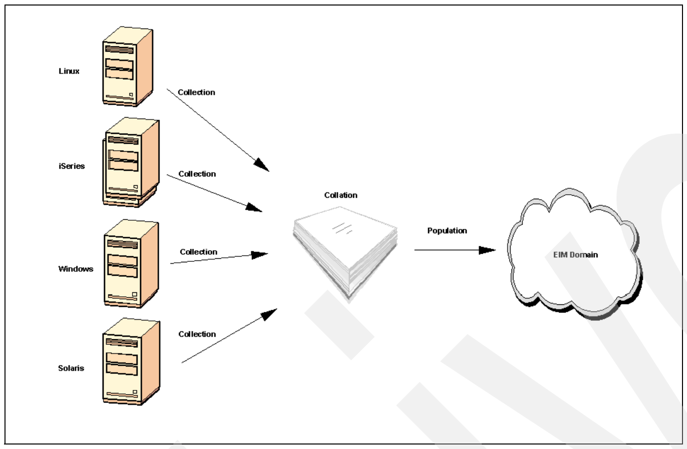


Figure 6-14 Three steps to success

6.5.1 Collection

Before users can be inputted into EIM first they must be identified. Users will have user names on many systems and all of these systems will need to be collected from. EIM provides a matching of all a principals user names to a master record. Remember from 6.3.3, “EIM identifiers” on page 76 and 6.3.5, “EIM associations” on page 81, these user names are called associations and the master record is called the EIM identifier. Depending on the different types of systems involved there are varied ways of collecting lists of user IDs this itself poses a problem to the systems administrator.

Problems associated with collection are:

- The number of systems that user IDs need to be collected from.
- The number of user IDs in these systems that need to be collected.
- The different types of systems pose a problem through the different ways user names are stored in the different operating systems. For example, in a Windows Domain, a Linux /etc/passwd or iSeries user profiles. The information held may not even be computerized and it is possible that some information will be held in employees personnel records.

6.5.2 Collation

Once a list of user names and the servers they relate to has been created there is a new task facing the administrator. This is which user names are related to each user in the organization?

Problems associated with collation are:

- The volume of user names to be consolidated.

- The possible number of matches available. For example, even on two systems with 20 users there are 380 potential combinations of user names. Even if we just add another system into this example, with 3 systems and 20 users there are 6840 combinations, although only 20 are correct. This number increases phenomenally depending on the combination of systems and users.
- Confirming the identity of a user against its username. How do you know that Janice Doe is in fact DoeJ on a particular server? To do this there must be some sort of verification procedure, a managers sign off which will be able to confirm that an employees user names.
- In the case of users such as the system administrators or task IDs users will have more than one username on a particular system. How should these entries be added to the EIM Identifier for that person?
- Where should the information be stored? Information is being collected from a variety of sources but where is it all going to be put in order to be collated.

6.5.3 Population

This is the final hurdle for the system administrator. The system administrator now has a list of user names and who they relate to. The new problem is how do you populate EIM with this information.

Problems associated with population are:

- The time associated with inputting entries into the EIM domain. For a pilot implementation of EIM the time it takes could be justified but for a large number of users time becomes an important factor.
- The cost to the organization of this time in performing data entry in which time could be spent more constructively with more important administrative tasks.

Suggestions for better collation

As we can see there are many problems posed through the collection, collation and population steps. This section describes how some of these more serious problems could be overcome:

- Hierarchies can be used to provide a structured ordering of information that could help in a matching process. For example, assume that we have collected all the user names for users in the Boston sales department. The server that these people access is defined at the regional level at a branch headquarters, the Northeast. Using a *taxonomy,* an ordered hierarchical list, about regions and locations therein we could investigate this taxonomy to see that Boston is actually in the Northeast section of the country. Assuming relevant information about this username and the server has been entered into a matching system more accurate matches on username to user can be made.
- Introduce a master list. Somewhere there must be a list or database that holds all of the people that are in an organization. An example of this list could be from human resources that lists all the people employed by the organization or a list of all employees in the payroll system. The advantage of introducing a master list is that it greatly simplifies the matching process. Rather than comparing lists of IDs for similar matches, the user namesuser names can be compared against the users in a master list to see to which user a username is most likely going to correspond.

There are a growing number of tools becoming available to help you with populating your EIM configuration. See Appendix E, “Available EIM products” on page 243 for a sample of some of the programs currently available from independent software vendors (ISV).

6.6 EIM User Management

In many organizations there are procedures for dealing with changing and updating the status of users/employees. We have put together some possible solutions that may help you plan for common business situations. Some of these situations may include:

- Employees leaving/Disabling users
- Users changing names
- Changing roles
- Consolidated passwords

6.6.1 Disabling users

One task that all administrators will have to accomplish will be dealing with users who are leaving. A recommendation when dealing with user names and user IDs is not to delete them but rather disable them. This has the advantage that the account is inactive and cannot be used. Another advantage is that if in the future another user with the same name starts at the company then that user does not inherit inadvertently any privileges from another user. When this new user is added it would be seen that there was already this user ID in existence on the system previously and a new ID would be created unique to the system.

The solution to the problem is using the different association types EIM allows us to enter when we add an association to an Identifier. The normal use of associations is to map between different user names using the source and target attributes. The third association type is *administrative*. Administrative associations are not returned by EIM lookup operations. Remember that there are two types of lookup operations with administrative IDs which do not yield results, these are:

- Querying for a user name for a system using an administrative association. It will not return anything even if an association exists.
- Querying for a user name on a system which is an administrative association. This query will not return anything either.

We can now see that if a user leaves we could make all of their associations administrative. This would prevent access to any system through EIM. Remember that the user would also need to be disabled in the Kerberos realm or in the case of Windows servers disabled in the Active Directory. Just because the identifier in EIM will not return a result does not mean access has been controlled to the computers listed therein. Each computer which has an associated entry in EIM for that user must be accessed by an administrator and the user account for that person on the system disabled, otherwise the user could physically go to that system and log on directly to that computer.

6.6.2 Users changing names

Another common situation in an organization is that of users changing names. A common example of this is if someone gets married. Depending on the organization, all user names could be changed and the old user names disabled. With EIM this process is changed from a large administrative request into a simple one time operation.

EIM Identifiers have extra fields associated with them. These fields are called *aliases* and *addition information*. The additional information field is an array of strings which can hold any user definable information required. The alias field allows for another users identity on the same EIM identifier.

In this example an alias would be placed on the existing users identifier to allow for the new name. This would not impact any of the systems which this user accesses. The user names in

EIM are user names which a user will not have to retype in order access services on these computers, therefore the user does not explicitly use these IDs and in effect they don't need to be changed. Any future IDs could be created using the new name if the user so wanted. The only IDs which could perhaps be impacted are the source IDs which the user logs onto the systems using.

6.6.3 Changing roles

This administrative request is actually two administrative requests in one. The first one we have covered is disabling the user IDs for systems which this user no longer needs to access, in EIM as well as the user accounts on the computers themselves. The second is to add the associations for the new systems which the user now needs to access. For information on how to add user identities into EIM go to 7.2.3, “Using iSeries Navigator to add identifiers and associations” on page 117.

6.6.4 Consolidated passwords

Assuming all applications in an organization are kerberized and EIM enabled there would only be one password per user. This would be the users password in Active Directory which the user would initially sign on with. Section 1.2, “Vertical versus horizontal SSO” on page 6 talks about the different directions of single signon and how that true vertical single signon is extremely hard to achieve. Many applications will be able to be single signon enabled or the workflow inside that application that access other systems will be single signon enabled. The number of passwords that each user has on systems now will be reduced, users only needing passwords to sign on with initially and for any applications which are not single signon enabled. This will lead to a decrease in the number of password resets that an administrator will have to perform as users will have less passwords to remember.

6.7 EIM server management situations

In this section we explain some possible ways that server aliases could be used to help with the process of EIM and servers.

The scenarios we are going to detail are:

- Clustered servers
- Server migration and consolidation
- Application registries

6.7.1 Clustered servers

In an organization it is very likely that, for backup or performance considerations, servers are clustered. The wide availability of different server types having clustering capability from operating system clustering to application server clustering this is a situation that may have to be considered. It is very common that these clustered servers are synchronized. A problem in EIM is that each of these clustered servers will have an registry associated with it. The question is how do you synchronize these application registries which sit in the domain outside of this cluster synching functions.

In the aliases types offered by iSeries Access this would not be able to be performed but it is possible using the EIM API's or through adding your own alias through iSeries Access. It is likely that after a cluster has been created there is a name or reference that represents these clustered units. An alias of “ClusterReference” could be used along with the name for these clustered units, detailing that a server is a part of this cluster. This alias could be used to retrieve a list of servers in a certain cluster and hence some synchronizing capability could be performed on the registries therein.

6.7.2 Server migration and consolidation

Another common procedure for administrators is that of server migration. This could be in the form of upgrading to a new more powerful server or in the case of business mergers consolidation so that employees use a central server rather than different servers of each section of the company. Registry aliases allow us a good solution to this problem. If a server is to be migrated from to a more powerful server there are many steps which administrator would follow. The administrator will have to migrate many things, applications, user profiles, services to name just a few. After this migration it is possible that the new server will have different name from the old server so that they can still be run in parallel for some time to make sure the migration has been a success. The registry alias property allows for registries on this old server to have an alias of the new server. This has the advantage that if this old server is removed from the system then the new server will be able to take over and assuming all user profiles and access rights are the same the EIM infrastructure should be able to be used.

6.7.3 Application registries and user groups

When developers are creating applications there is usually some sort of user configuration section to detail security information or access levels about users. This user registry could be moved into EIM by creating an application specific alias for a particular registry. The administrator and the developer would converse to assign this alias to a registry. The registry would be retrieved using the getRegistryByAlias method which takes in a registry alias object parameter.

In some applications access to an application is defined by the level of security of a particular group. In addition to using this user alias we could add a piece of additional information to each EIM identifier to dictate the level of access a user has for a particular application. For example if a registry *applicationOne* was created in EIM then this could be added to each identifier of the user who wanted access to applicationOne. When the getRegistryByAlias method is used it returns the registry from which the registry users can be extracted. From these users identifiers can be found and this additional information retrieved. This additional information could be something similar to *applicationOne:admin* which would determine the level of access this user has.

There would still be the need for a configuration section for security for each user in the database but the user name with which the user comes in from would not be a problem as EIM would solve this for the user.

Registry aliases in this context allow application developers to write applications without having to know which registry their application will use, an alias can just be added at a later date. This has a bid advantage for commercial software where is it highly unlikely that registries will have the same name and aliases allow for the quick integration of an application into EIM.

Part 3

Installation and configuration

Part 3 provides you with the “how to” required to enable a single signon environment, complete with Network Authentication Service and Enterprise Identity Mapping on the iSeries. To make your configuration go smoothly, we recommend that you complete the worksheets provided in Appendix D, “Planning forms” on page 239 prior to starting the configuration.

7

Enabling Network Authentication Service and Enterprise Identity Mapping

In this chapter we show, in detail, how to enable and set up all the components needed to establish the base for our single signon scenario, which was described in Chapter 3, “The redbook example scenario” on page 29.

7.1 Configure Network Authentication Service

To help insure success, make sure that you have met all the prerequisites listed in Chapter 2, "Planning for Network Authentication Service and Enterprise Identity Mapping implementation" on page 15. We have provided blank worksheets in Appendix D, "Planning forms" that will assist you in verifying your readiness and in preparing for the implementation of single signon (SSO). Table D-1 on page 240 is a checklist that will help you determine that you have met all the required prerequisites. Table D-2 on page 241 is a worksheet that we designed to help you fill in the fields that are presented in the different wizards and configuration screens. You can refer back to 2.4, "Information to collect before you start" on page 24, to get instructions on how to find each of the parameters.

Table 7-1 shows how our worksheet looked prior to the start of our configuration.

Table 7-1 Our scenario's worksheet

Item	Information to collect	Result
A	What is the name of the Kerberos default realm to which the iSeries will belong?	ITSO.IBM.COM
B	What is the KDC for this Kerberos default realm?	KrbSvr2000
C	What is your KDC's fully qualified host name?	KrbSvr2000.itso.ibm.com
D	What is the port on which the KDC listens?	88
E	What is name of the password server for this KDC?	KrbSvr2000
F	What is the port of your password server?	464
G	What is the password for your iSeries service principal(s)?	win4IBM
The following items will be used to create the iSeries principal on the KDC		
H	What is the name of the Kerberos principal?	krbsvr400 (when creating the iSeries principal this name must be used)
I	What is your iSeries host name?	as20
J	What is the fully qualified host name of the iSeries?	as20.itso.ibm.com
K	What is the name of the Kerberos default realm to which the iSeries server belongs? (Default: domain name converted to upper case)	ITSO.IBM.COM
L	What is the full name of the Kerberos principal? (`krbsvr400/fully.qualified.host.name@YOUR.KERBEROS.REALM`)	krbsvr400/as20.itso.ibm.com@ ITSO.IBM.COM
M	What is the password / shared secret for this principal? (Must be the same as item G)	win4IBM
The following items will be used to configure Enterprise Identity Mapping (EIM)		
N	Which type of basic EIM configuration do you want to create on your iSeries system? ► Join an existing domain ► Create and join new domain	Create and join new domain
O	Where do you want to configure your EIM domain, or what EIM domain you want to join?	as20.itso.ibm.com

Item	Information to collect	Result
P	What is the name of the EIM domain you want to create or join?	ITSO EIM
Q	Do you want to specify a parent DN for the EIM domain? If yes - specify the parent DN	NO
R	What is the administrator distinguished name (DN) on the LDAP server which will be used as the EIM domain controller?	CN=administrator (see note in section titled “Filling in the Worksheet” on page 26)
S	What is the administrator password on the LDAP server will be used as the EIM domain controller?	ldappw

After you have confirmed that the Active Directory (which is described in 2.2, “Required network components” on page 17) has been set up and its KDC is running, you can start Network Authentication Service configuration. This includes the following steps:

- Running the iSeries Navigator wizard to set up Network Authentication Service on your iSeries.
- Creating Kerberos principal for your iSeries system in the KDC of your Windows 2000 server.
- Using the Qshell environment on your iSeries system to verify that the Network Authentication Service is configured correctly.

7.1.1 Setting up Network Authentication Service with iSeries Navigator wizard

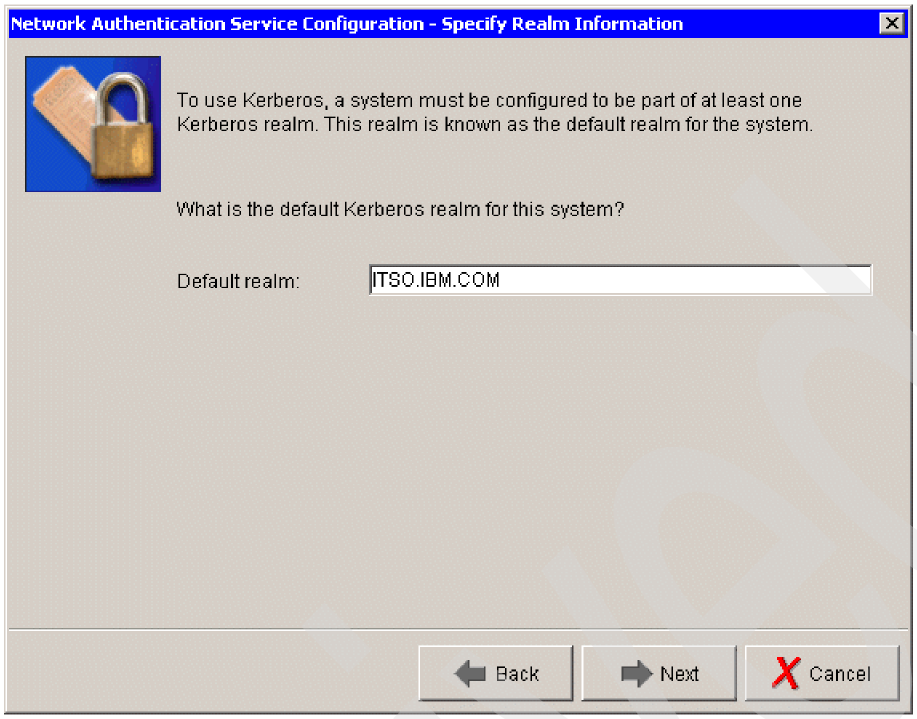
To start the iSeries Navigator’s Network Authentication Service wizard perform the following steps:

1. In the left pane of the iSeries Navigator expand your **iSeries system -> Security** then right-click **Network Authentication Service** and select **Configure.**

   Note: If you see **Reconfigure** option instead of **Configure**, it indicates that the Network Authentication Service has already been configured. You can either reconfigure it (the re configuration proceeds much in a similar way as the configuration), or you can right-click Network Authentication Service and select **Properties** to verify the current configuration.

2. Network Authentication Service configuration welcome window appears; read the information and click **Next**.
3. In the Specify Realm Information window (Figure 7-1), enter the name of the realm that will serve as you default Kerberos realm. This is item A in Table D-2 on page 241. Click **Next**.

   Note: The Windows 2000 KDC is case sensitive and the name of the realm is always in upper case. Under standard conventions the name of the realm is the domain name, converted to uppercase.


Network Authentication Service Configuration - Specify Realm Information

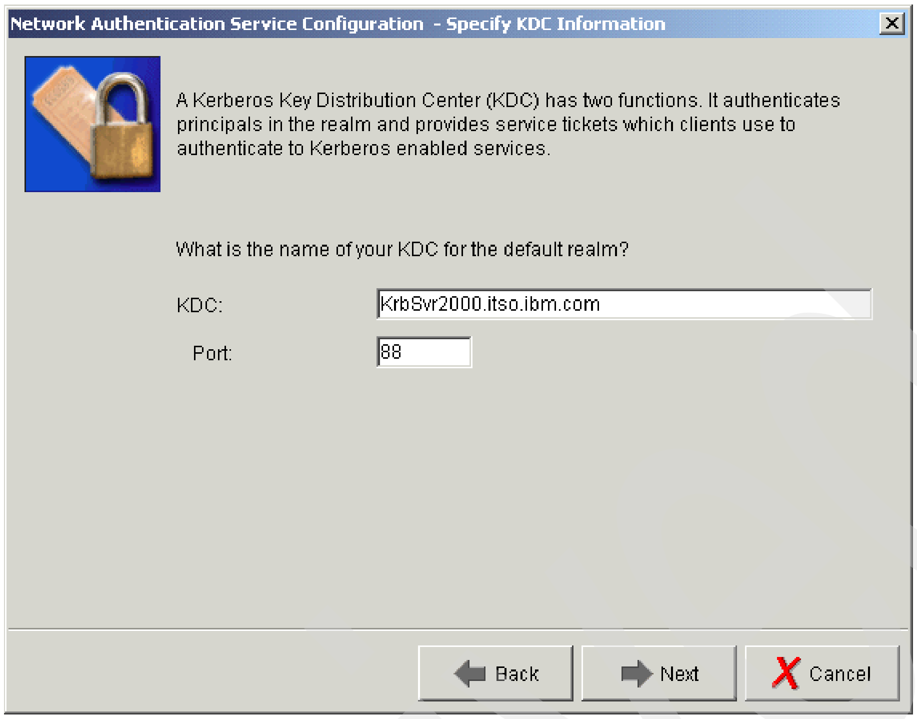
To use Kerberos, a system must be configured to be part of at least one Kerberos realm. This realm is known as the default realm for the system.

What is the default Kerberos realm for this system?

Default realm: ITSO.IBM.COM

Figure 7-1 Network Authentication Service Configuration: specify realm information

4. In the Specify KDC Information window, Figure 7-2, enter the following information:
   - Specify the fully qualified host and domain name of your Kerberos Key Distribution Center (KDC). This is item C in Table D-2 on page 241.
   - The default port for Kerberos is 88. Unless your KDC has been configured to listen on on a port other than the default this parameter does not need to be changed. Item D in Table D-2 on page 241 is the value that should be entered.
   - Click **Next**.


Network Authentication Service Configuration - Specify KDC Information

A Kerberos Key Distribution Center (KDC) has two functions. It authenticates principals in the realm and provides service tickets which clients use to authenticate to Kerberos enabled services.

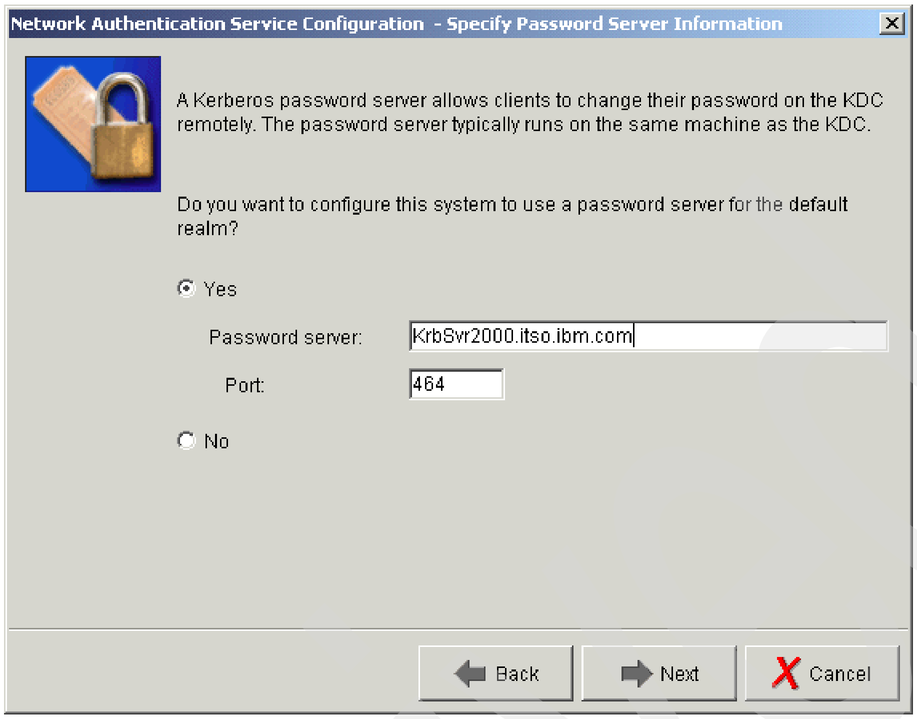
What is the name of your KDC for the default realm?

KDC: KrbSvr2000.itso.ibm.com

Port: 88

Figure 7-2 Network Authentication Service Configuration: specify KDC information

5. The next window gives you the option to allow principals to change the Kerberos passwords remotely. If you want to allow clients to change passwords select **Yes** in the Specify Password Server Information window, Figure 7-3. After selecting yes you may fill in the required fields:
   - Specify the fully qualified name of your Windows 2000 server as the name of password server, item E in Table D-2 on page 241.
   - Leave the default port number 464 unchanged, unless you know that your Windows 2000 server has been configured differently. This is item F in Table D-2 on page 241.
   - Click **Next**.


Network Authentication Service Configuration - Specify Password Server Information

A Kerberos password server allows clients to change their password on the KDC remotely. The password server typically runs on the same machine as the KDC.

Do you want to configure this system to use a password server for the default realm?

[✓ Yes]

Password server: KrbSvr2000.itso.ibm.com

Port: 464

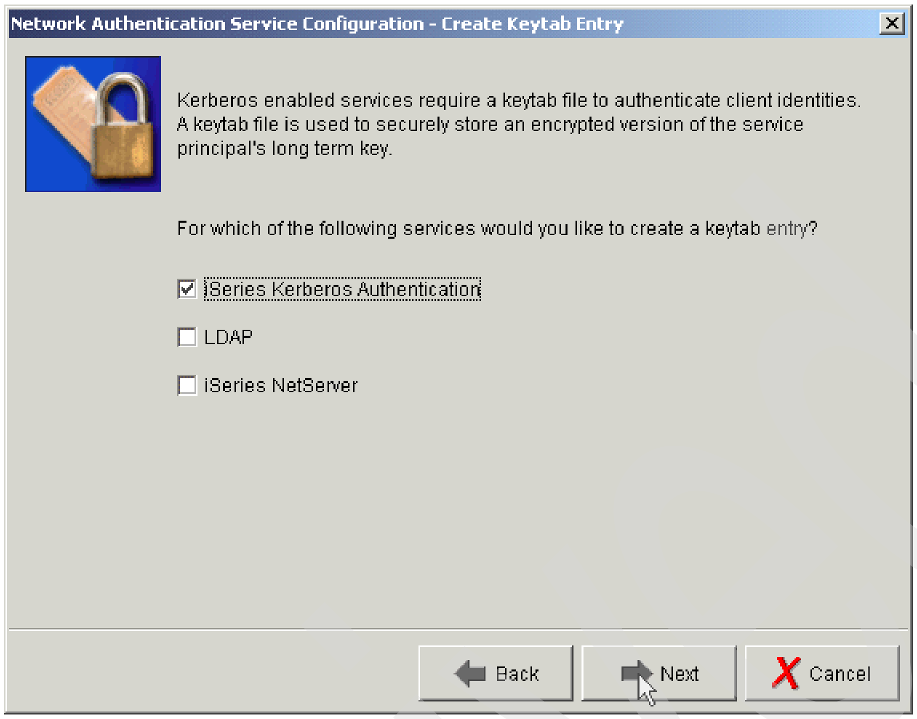
No

Figure 7-3 Network Authentication Service Configuration: specify password server information

6. The Create Keytab Entry window (Figure 7-4) lets you select which services on your iSeries the wizard should enable for Kerberos authentication. In our scenario we only enable the iSeries Kerberos Authentication, as shown in Figure 7-4. This enables for Kerberos nearly all OS/400 services, excluding LDAP and NetServer.

Note: Both LDAP and iSeries NetServer can use Kerberos tickets to authenticate but currently only NetServer can take advantage of EIM. If your current plans are to enable Netserver and LDAP for Kerberos authentication it may save you some time to check the boxes now. We do not follow that path in this walk through but there is only a slight difference in the path the wizard follows. After checking the additional boxes you will be presented two additional windows that will prompt you for the passwords for the LDAP principal and the NetServer principal. Be sure you document those passwords because you will need to supply them in the KDC configurations. If you choose not to enable LDAP and Netserver at this time, then later in 8.4, “Enabling NetServer for single signon” on page 153 we demonstrate how to enable NetServer for Kerberos authentication and EIM.

Select the **iSeries Kerberos Authentication** check box and click **Next**.


Network Authentication Service Configuration - Create Keytab Entry

Kerberos enabled services require a keytab file to authenticate client identities. A keytab file is used to securely store an encrypted version of the service principal's long term key.

For which of the following services would you like to create a keytab entry?

[✓] iSeries Kerberos Authentication

[ ] LDAP

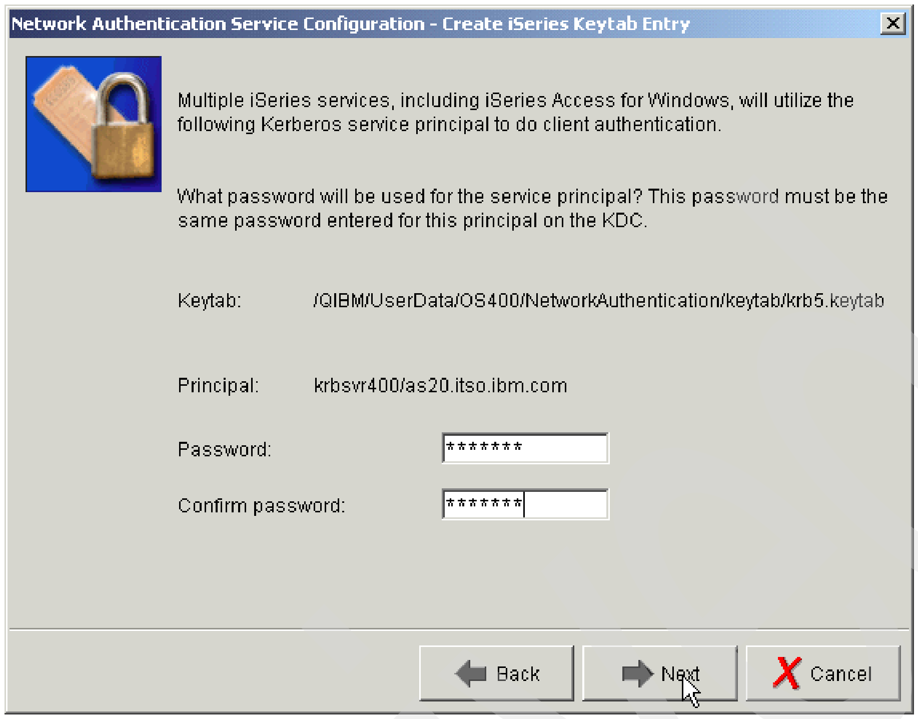
[ ] iSeries NetServer

Figure 7-4 Network Authentication Service Configuration: create keytab entry

7. In the Create iSeries Keytab Entry window (Figure 7-5) you specify the password for the Kerberos principal, that is, *shared secret*. This is item G in Table D-2 on page 241. You have to enter it twice for verification as it is not displayed.

Note: The Create iSeries Keytab Entry window in Figure 7-5 shows two important items:

- The path to the keytab file which the wizard is to generate (next to *Keytab:* label).
- Name of the corresponding Kerberos principal in the KDC (next to *Principal:* label).


Network Authentication Service Configuration - Create iSeries Keytab Entry

Multiple iSeries services, including iSeries Access for Windows, will utilize the following Kerberos service principal to do client authentication.

What password will be used for the service principal? This password must be the same password entered for this principal on the KDC.

Keytab: /QIBM/UserData/OS400/NetworkAuthentication/keytab/krb5.keytab

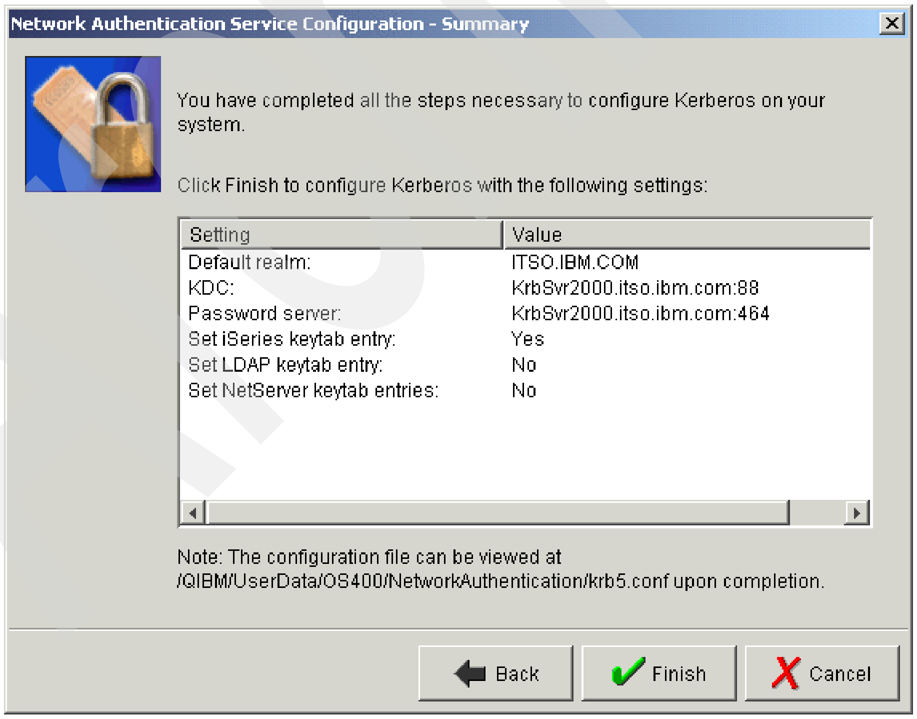
Principal: krbsvr400/as20.itso.ibm.com

Password: *******

Confirm password: *******

Figure 7-5 Network Authentication Service Configuration - create iSeries keytab entry

8. The final step is to verify the entries in the Configuration Summary (Figure 7-6). Use the back button to go back and make any changes that my be necessary. Then click **Finish**.



You have completed all the steps necessary to configure Kerberos on your system.

Click Finish to configure Kerberos with the following settings:

Setting	Value
Default realm:	ITSO.IBM.COM
KDC:	KrbSvr2000.itso.ibm.com:88
Password server:	KrbSvr2000.itso.ibm.com:464
Set iSeries keytab entry:	Yes
Set LDAP keytab entry:	No
Set NetServer keytab entries:	No

Note: The configuration file can be viewed at /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf upon completion.

Figure 7-6 Network Authentication Service configuration summary

Your new realm should now be seen in the right pane of iSeries Navigator. You may need to refresh the contents of the window using the F5 key.

7.1.2 Create Kerberos principal for your iSeries server

Your next task is to create a Kerberos principal for your iSeries system. This will allow the iSeries to take part in the Kerberos realm for the authentication of the users.

You perform these steps:

- Create Active Directory account for your iSeries system
- Associate the account with the Kerberos principal using the `ktpass` command
- Check and complete the Active Directory account

Creating Active Directory account for your iSeries server

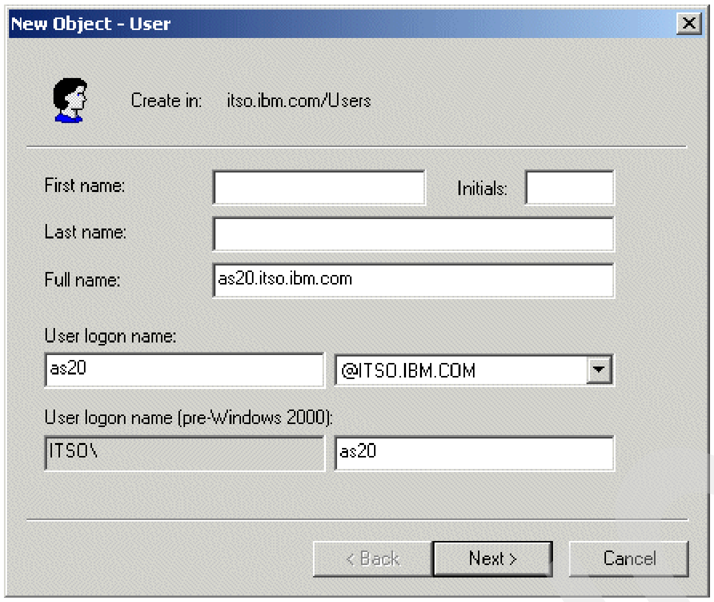
First you have to create a user account for your iSeries system in the Active Directory.

1. At the console of your Windows 2000 server, click **Start -> Programs -> Administrative Tools -> Active Directory Users and Computers**. This opens the Active Directory Users and Computers window.
2. Right-click the **Users** folder in the left pane and select **New -> User**. This opens the New Object - User window.
3. In the New Object - User window (Figure 7-7), enter the name of the Active Directory account you are creating (item J in the Kerberos configuration worksheet). Though this name may be arbitrary, we recommend that you use the name of your iSeries system.
   - Enter this name in the Full name field.
   - Enter item I from your configuration worksheet in the User logon name field; it is automatically copied in the pre-Windows 2000 name field.

   The name of the account (here `as20`) may be entered in lower case, upper case or mixed case as desired. This holds true for the left part of the User logon name also, the `ktpass` command will change it later anyway.You don’t need to fill in the First name, Last name and Initials.
4. Leave the default (here `@ITSO.IBM.COM`) in the combo box on the right of the User logon name. It is the default Kerberos realm name for the Active Directory, prefixed with the @ sign. By default it is the domain name of the server *converted to upper case*.

   Important: The Windows 2000 KDC is case sensitive and the name of the Kerberos realm is, by convention, is in upper case.

   Click **Next** to proceed to next window.



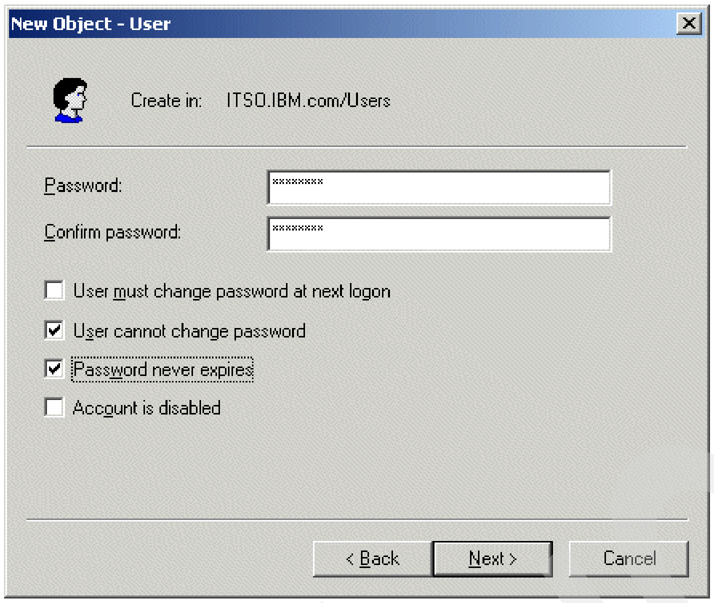
First name:
Initials:
Last name:
Full name: as20.itso.ibm.com
User logon name: as20 @ITSO.IBM.COM
User logon name (pre-Windows 2000): ITSO\ as20

Figure 7-7 Enter the name of the Active Directory account

5. Another New Object - User window appears (Figure 7-8). In this window you are prompted to enter the password and make several password policy decisions. The password is the *Kerberos shared secret* you already entered in the Network Authentication Service configuration wizard, item M in Table D-2 on page 241. You have to enter it twice for verification as it is not displayed.

Suggestion: Depending on your internal policies for passwords you may want to check the **User cannot change password** and **Password never expires** attributes to prevent the account getting locked either by human error or by the password change policy being applied to it.

Click **Next**.


New Object - User

Create in: ITSO.IBM.com/Users

Password: *********

Confirm password: *********

- [ ] User must change password at next logon
- [x] User cannot change password
- [x] Password never expires
- [ ] Account is disabled

Figure 7-8 Fill in the Kerberos shared secret

6. Click **Finish**.

Associate the account with the Kerberos principal

Unlike the Windows clients, the Network Authentication Service on iSeries system uses Unix-type services of the Kerberos implementation in Active Directory. You will need to use the **`ktpass`** command to modify the Active Directory account and tie it to the name of the Kerberos principal.

Note: The **`ktpass`** command is a part of the Support Tools, included on the Windows 2000 Server installation CD-ROM. If the **`ktpass`** command is missing from your Windows 2000 server, refer to Appendix C, "Windows 2000 Kerberos tools".

7. Open the Windows 2000 server command window and enter the `ktpass` command as shown in Figure 7-9. You will substitute your own values, from Table D-2 that you completed, for the following parameters:

   `-princ` = item L

   `-mapuser` = item I

   `-pass` = item M

Note: `-mapOp` set is optional. This option overwrites the existing principal mapping with the new one (useful when you think you made a mistake). The Op portion of the command is an upper case o, not a zero.

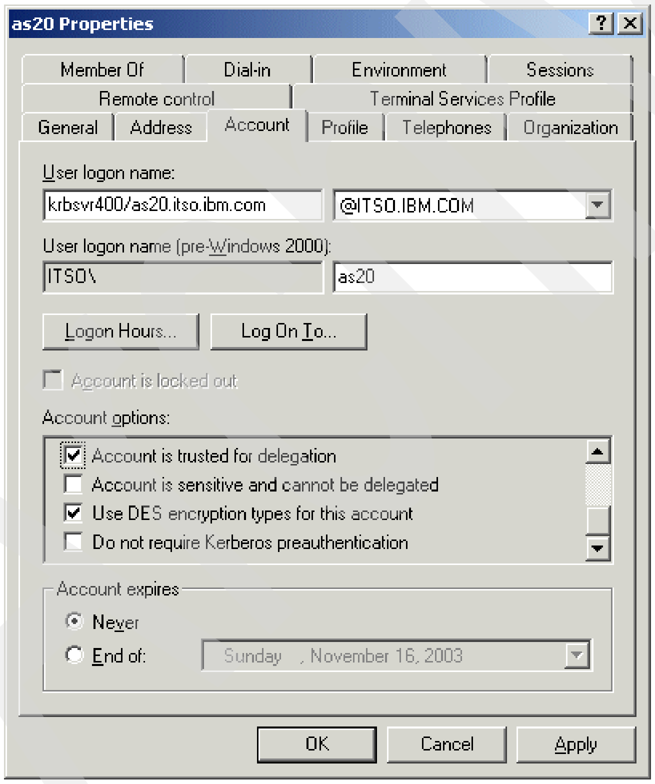
```
C:\>ktpass -princ krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM -mapuser as20 -pass win4IBM
-mapOp set

Successfully mapped krbsvr400/as20.itso.ibm.com to as20.
Key created.
Account has been set for DES-only encryption.
C:\>
```

Figure 7-9 Entering the ktpass command

Check and complete the Active Directory account

In the right pane of the Active Directory Users and Computers window, you can right-click the account and select **Properties**. In the window that opens, click the Account tab; you can see the changes made by the **ktpass** command. An example of the iSeries account properties is shown in Figure 7-10.

8. In the Account options window scroll down until you see the option *Account is trusted for delegation*, you need to check this box associated with this option. Trusting the account for delegation allows the iSeries system to forward its tickets to other services like `QFileSrv.400`, `DRDA` and `PC5250`. Service principals must be trusted for delegation for single signon. Click OK; you have now completed the Windows portion of the setup.



User logon name: krbsvr400/as20.itso.ibm.com @ITSO.IBM.COM

User logon name (pre-Windows 2000): ITSO\ as20

☐ Account is locked out

Account options:

☑ Account is trusted for delegation
☐ Account is sensitive and cannot be delegated
☑ Use DES encryption types for this account
☐ Do not require Kerberos preauthentication

Account expires
[✓ Never]
End of: Sunday , November 16, 2003

Figure 7-10 iSeries Kerberos Account properties

7.1.3 Verify Network Authentication Service setup

You have completed two steps needed to implement Network Authentication Service authentication of users on your iSeries system:

- Network Authentication Service is now set up on the iSeries system

- Kerberos principal is created for your iSeries system in the KDC of your Windows 2000 server.

You should now check that these two components co-operate. The steps described in this chapter are not required for the Network Authentication Service to work. However, by performing these steps you confirm that the Kerberos environment is working correctly.

Note: To verify the configuration, you will need to create a directory on the iSeries for the krbsvr400 principal. On the iSeries command line issue the command:

```
crtdir '/home/krbsvr400'
```

This directory is used to store the Kerberos credentials after the kinit command is complete.

1. Start the Qshell interpreter in an 5250 session to your iSeries system, by entering OS/400 command:

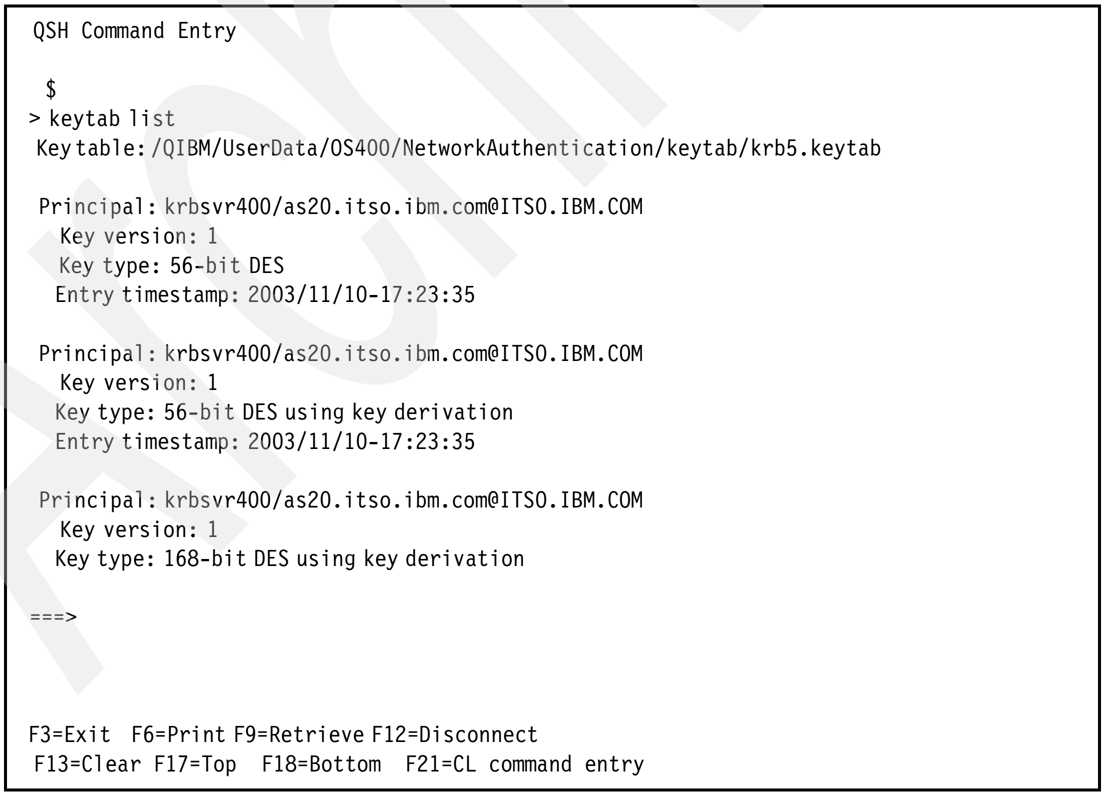
   ```
   QSH
   ```

2. Use the following Qshell command to list the current keys in the Kerberos key table:

   ```
   keytab list
   ```

   An example of the output is shown in Figure 7-11.

   If the wizard completed correctly and made contact with the KDC, the key table should contain three entries for the `krbsvr400` principal (at different encryption levels). If you don't see the entries, or they appear to be incorrect refer to Appendix B, "Troubleshooting".



```
 QSH Command Entry

  $
> keytab list
 Key table: /QIBM/UserData/OS400/NetworkAuthentication/keytab/krb5.keytab

 Principal: krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM
   Key version: 1
   Key type: 56-bit DES
   Entry timestamp: 2003/11/10-17:23:35

 Principal: krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM
   Key version: 1
   Key type: 56-bit DES using key derivation
   Entry timestamp: 2003/11/10-17:23:35

 Principal: krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM
   Key version: 1
   Key type: 168-bit DES using key derivation

===>



F3=Exit  F6=Print F9=Retrieve F12=Disconnect
 F13=Clear F17=Top  F18=Bottom  F21=CL command entry
```

Figure 7-11 List the Kerberos keytab entries

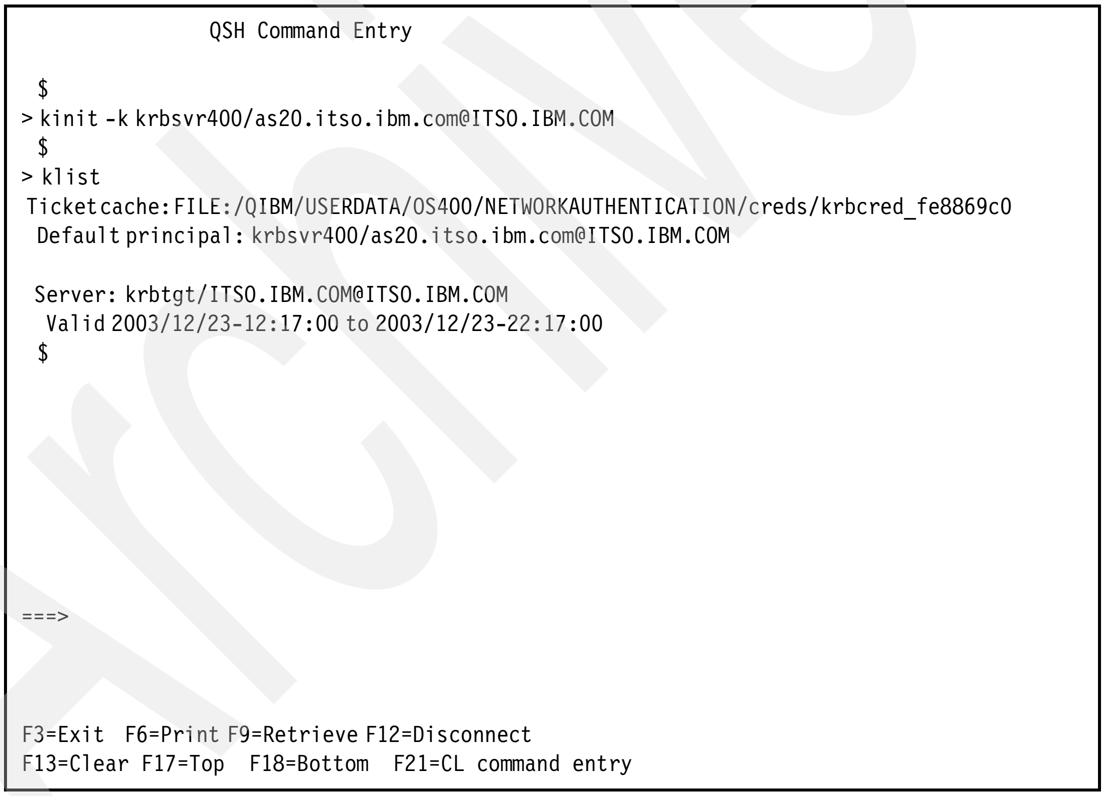
3. Once the key table is verified the next step is to request a ticket granting ticket (TGT) from the KDC. Use the **`kinit`** command from a Qshell command line as shown in Figure 7-12.

   > **Important:** All components of the principal name have to be entered in correct case. See "General TCP/IP considerations" on page 19 for a discussion of name resolution.
   >
   > - The service name `krbsvr400` has to be entered in lower case.
   > - The iSeries host name has to agree in case with the DNS entry or with the entry in the local host table on the client, if used.
   > - The name of the realm has to be entered in upper case.

   The **`kinit`** command should complete without any messages and return the command prompt.

4. List the Ticket Granting Ticket (TGT) using the **`klist`** command from Qshell as shown in the lower part of Figure 7-12.

If any of the Qshell commands ended with an error message, go to Appendix B, "Troubleshooting".



```
              QSH Command Entry

 $
> kinit -k krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM
 $
> klist
 Ticket cache: FILE:/QIBM/USERDATA/OS400/NETWORKAUTHENTICATION/creds/krbcred_fe8869c0
 Default principal: krbsvr400/as20.itso.ibm.com@ITSO.IBM.COM

 Server: krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
  Valid 2003/12/23-12:17:00 to 2003/12/23-22:17:00
 $








===>



F3=Exit  F6=Print F9=Retrieve F12=Disconnect
F13=Clear F17=Top  F18=Bottom  F21=CL command entry
```

Figure 7-12 Request TGT

This completes the verification of the Network Authentication Service setup.

7.2 Enable EIM

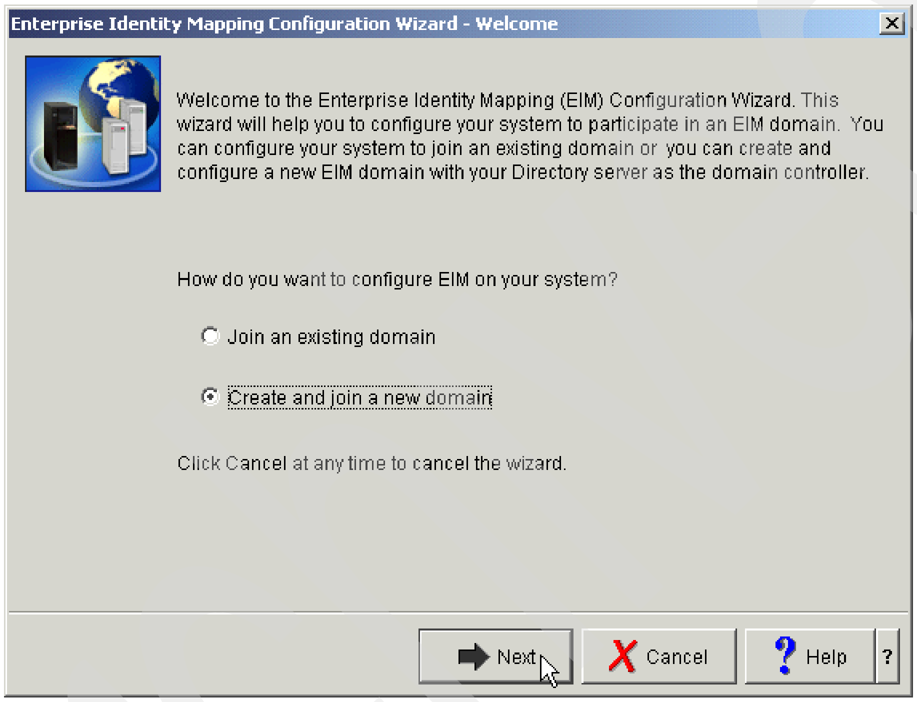
The following steps will enable Enterprise Identity Mapping (EIM):

- Configure EIM domain and EIM domain controller using the EIM configuration wizard.

- Add the EIM domain to the list of domains to be managed from your client.
- Use iSeries navigator to add a few users to the domain and associate them with their identities in user registries.

7.2.1 Using EIM configuration wizard

1. In iSeries Navigator expand the iSeries that you want to configure the EIM domain controller on. Then expand **Network** -> **Enterprise Identity Mapping**, right-click **Configuration** and the select **Configure**. In the EIM wizard Welcome window (Figure 7-13) select the option Create and join new domain.


Enterprise Identity Mapping Configuration Wizard - Welcome

Welcome to the Enterprise Identity Mapping (EIM) Configuration Wizard. This wizard will help you to configure your system to participate in an EIM domain. You can configure your system to join an existing domain or you can create and configure a new EIM domain with your Directory server as the domain controller.

How do you want to configure EIM on your system?

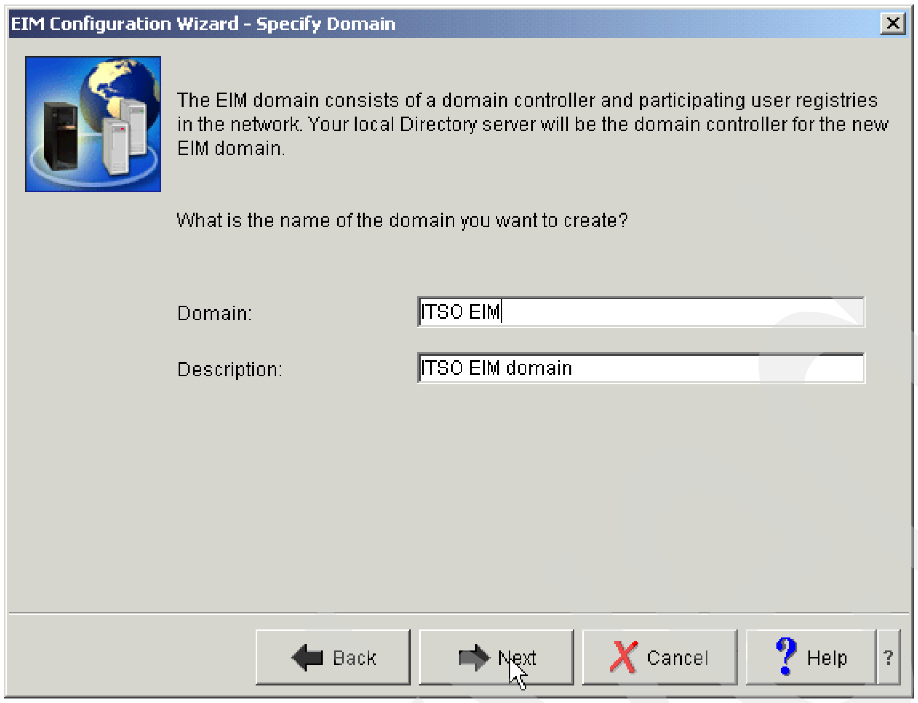
( ) Join an existing domain

(•) Create and join a new domain

Click Cancel at any time to cancel the wizard.

Figure 7-13 EIM configuration wizard - welcome

2. If your LDAP server is running while you are using the wizard a warning window will appear to inform you that the wizard will need to stop and restart the LDAP server. Click **Yes**.
3. In the Specify Domain window (Figure 7-14) enter the EIM domain name (item P from the Configuration worksheet) and description; click **Next**.


EIM Configuration Wizard - Specify Domain

The EIM domain consists of a domain controller and participating user registries in the network. Your local Directory server will be the domain controller for the new EIM domain.

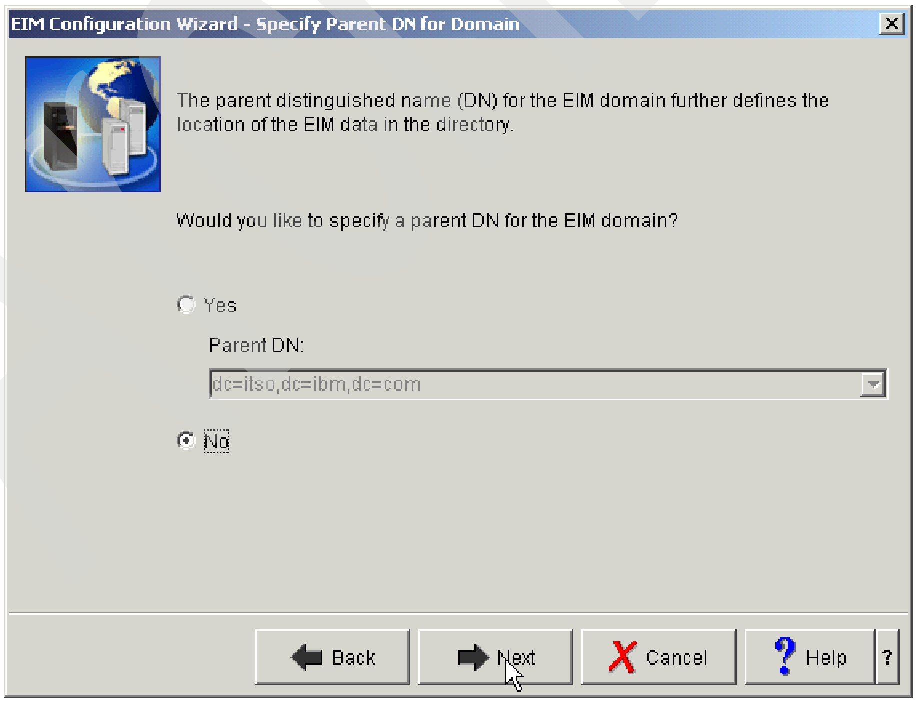
What is the name of the domain you want to create?

Domain: ITSO EIM

Description: ITSO EIM domain

Figure 7-14   EIM configuration wizard - specify domain

4. In the Specify Parent DN window (Figure 7-15) you can designate specific placement of the EIM domain sub-tree in the LDAP directory structure (item Q in the Configuration worksheet). To create the EIM domain controller as a separate tree in the LDAP structure select the **No** option; then click **Next**.


EIM Configuration Wizard - Specify Parent DN for Domain

The parent distinguished name (DN) for the EIM domain further defines the location of the EIM data in the directory.

Would you like to specify a parent DN for the EIM domain?

Yes [✓ No]

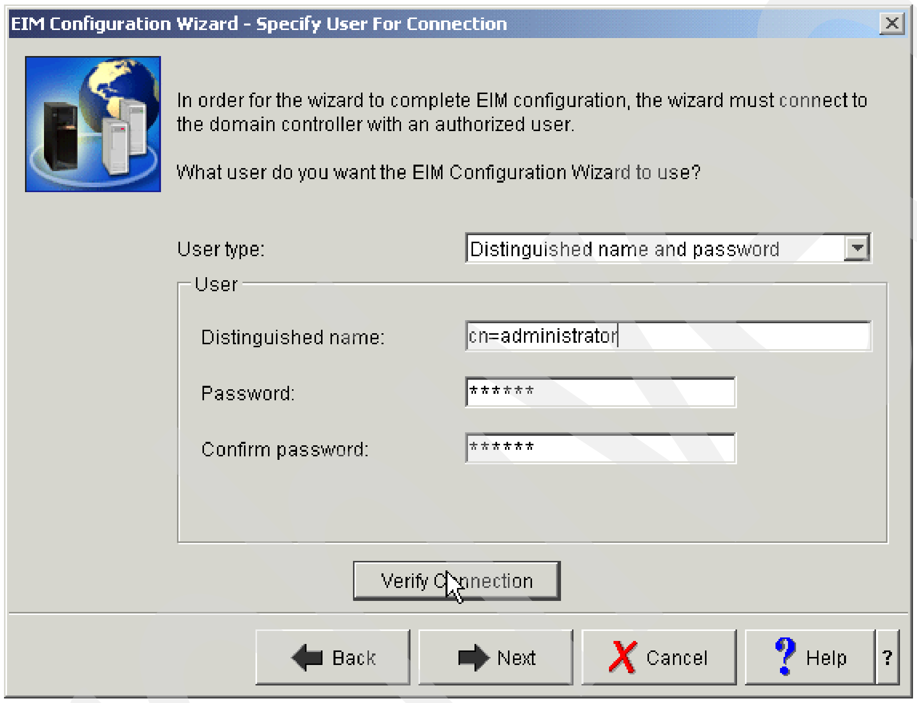
Parent DN: dc=itso,dc=ibm,dc=com

Figure 7-15   EIM configuration wizard - specify parent DN for domain

5. In the Specify User For Connection window (Figure 7-16) you specify which user the wizard uses when it connects to the LDAP server to perform the configuration task for you (this is just a one-time connection while the wizard is running).

   Use the distinguished name (DN) and password from item R and S in the configuration worksheet (we used the value `cn=administrator` as shown in Figure 7-16).

6. Click **Verify Connection** to verify the connection to the LDAP server. If the connection is successfully verified, click **OK** in the confirmation window and then **Next** in the Specify User For Connection window. If you are having a problem with the connection to the LDAP server see Appendix B, "Troubleshooting".


EIM Configuration Wizard - Specify User For Connection

In order for the wizard to complete EIM configuration, the wizard must connect to the domain controller with an authorized user.

What user do you want the EIM Configuration Wizard to use?

User type: Distinguished name and password

User

Distinguished name: cn=administrator

Password: ******

Confirm password: ******

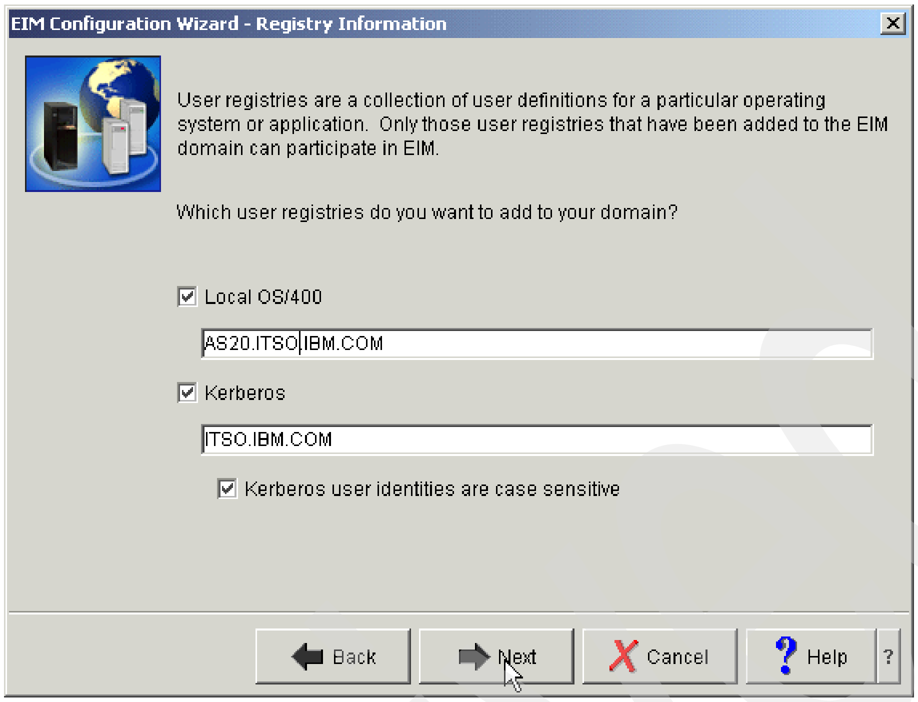
Figure 7-16 EIM configuration wizard - specify user for connection

7. In the Registry Information window (Figure 7-17) you may request two user registries to be automatically added to your domain; you should check both of them:

   – Local OS/400 registry for your iSeries system, hosting the EIM domain controller
   – Kerberos registry - in our case located on the Windows 2000 server

Tip: You may simplify your configuration and consequently any debugging that may need to be done if you de-select the *Kerberos user identities are case sensitive* check box.

Click **Next** to proceed.


EIM Configuration Wizard - Registry Information

User registries are a collection of user definitions for a particular operating system or application. Only those user registries that have been added to the EIM domain can participate in EIM.

Which user registries do you want to add to your domain?

☑ Local OS/400

AS20.ITSO.IBM.COM

☑ Kerberos

ITSO.IBM.COM

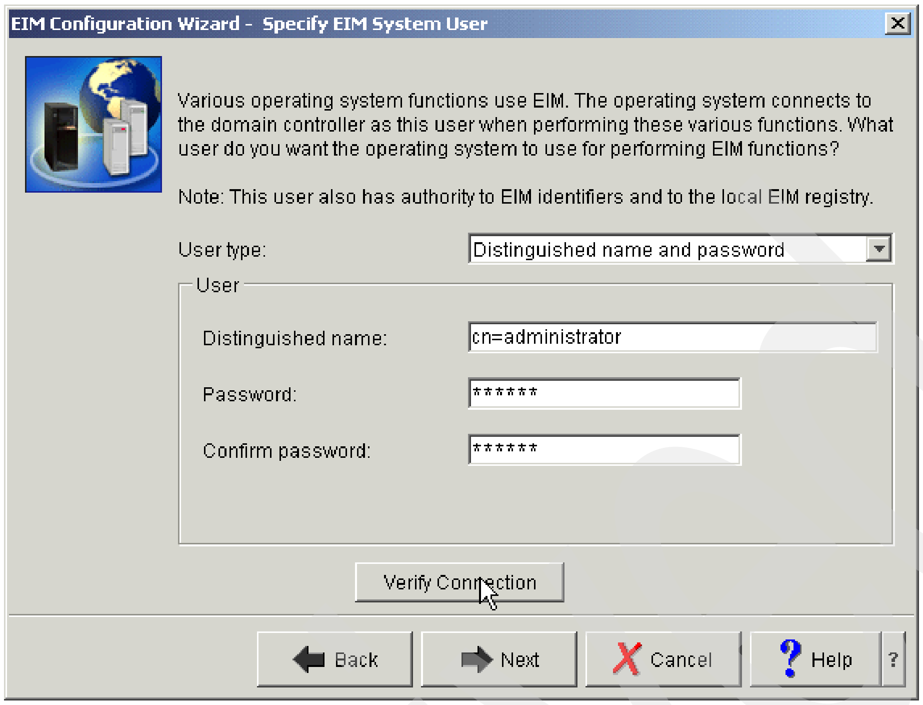
☑ Kerberos user identities are case sensitive

Figure 7-17 EIM configuration wizard - registry information

8. The Specify EIM System User window (Figure 7-18) is very similar to the Specify User For Connection window (step 6 on page 113 and Figure 7-16 on page 113). But now you specify an EIM System User, whose account is used to connect to the EIM domain controller by various operating system functions. This contrasts with the Specify User For Connection window, where you specified the user which is to be used currently by the EIM set-up wizard (only once).

   Again, we recommend that you create a new LDAP user and add it to the EIM Admin group. Use this new user ID and password for the system. Creating LDAP users is outside the scope of this redbook so we will use the LDAP administrator's ID and password instead.

   Verify the connection, as you did in Step 6 on page 113. Then click **Next** on the Specify EIM System User window.


EIM Configuration Wizard - Specify EIM System User

Various operating system functions use EIM. The operating system connects to the domain controller as this user when performing these various functions. What user do you want the operating system to use for performing EIM functions?

Note: This user also has authority to EIM identifiers and to the local EIM registry.

User type: Distinguished name and password

User

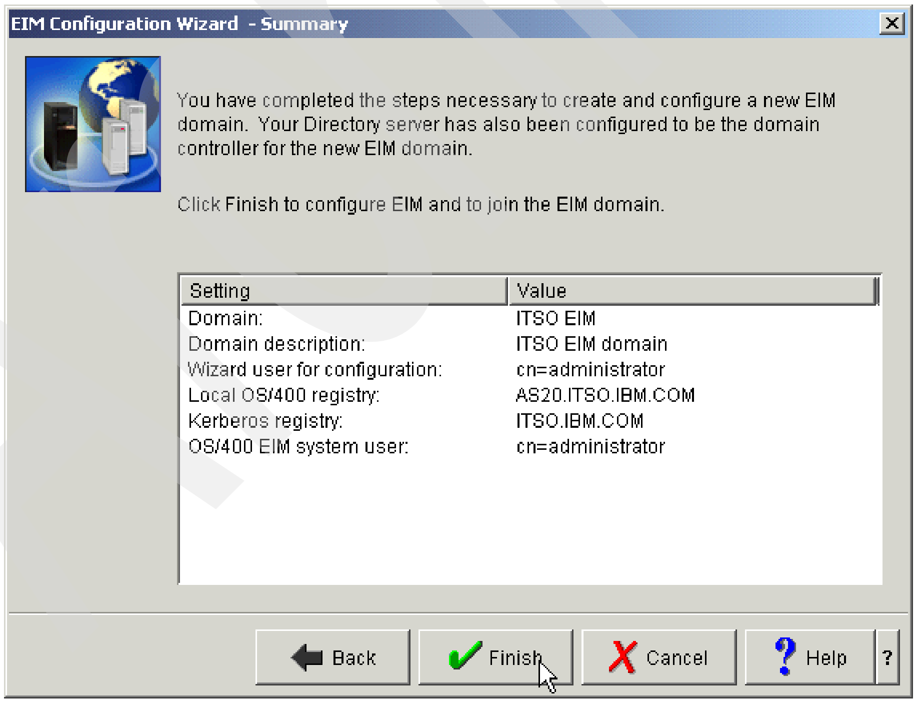
Distinguished name: cn=administrator

Password: ******

Confirm password: ******

Figure 7-18 EIM configuration wizard - specify EIM system user

9. The Summary window (Figure 7-19) recaps the settings you have specified. Verify that they are accurate and then click **Finish**.


EIM Configuration Wizard - Summary

You have completed the steps necessary to create and configure a new EIM domain. Your Directory server has also been configured to be the domain controller for the new EIM domain.

Click Finish to configure EIM and to join the EIM domain.

Setting	Value
Domain:	ITSO EIM
Domain description:	ITSO EIM domain
Wizard user for configuration:	cn=administrator
Local OS/400 registry:	AS20.ITSO.IBM.COM
Kerberos registry:	ITSO.IBM.COM
OS/400 EIM system user:	cn=administrator

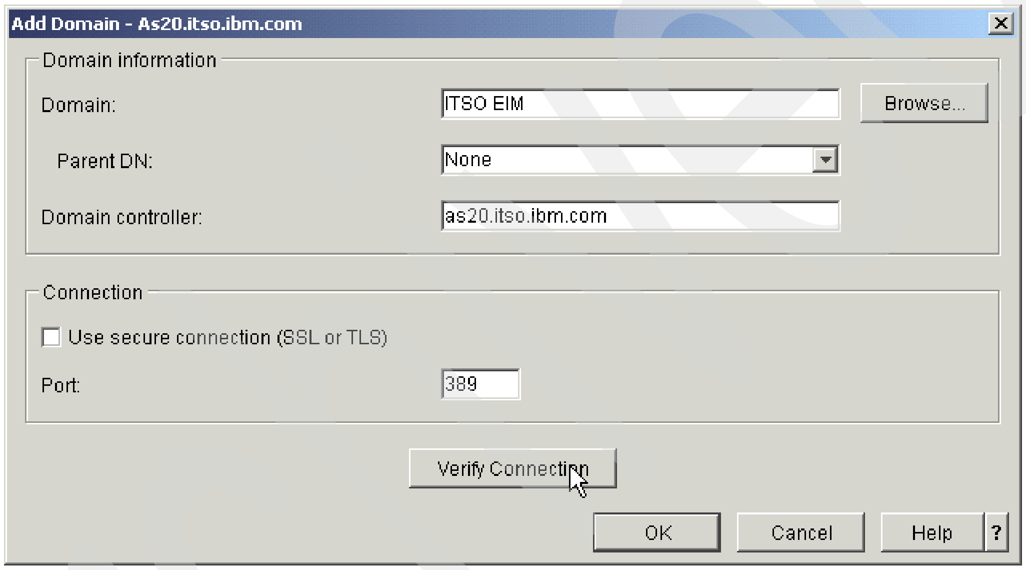
Figure 7-19 EIM configuration wizard - summary

7.2.2 Add the EIM domain to be managed

Now you need to add the EIM domain just configured to the list of EIM domains to be managed from your client (iSeries Navigator on your workstation).

Note: The domain management information is stored locally on the client. That is, if you use iSeries Navigator on another PC, you will have to add the domain for management again. Once the domain is added at the particular client, it should appear in its Domain Management.

1. In the iSeries Navigator, expand **Network** -> **Enterprise Identity Mapping**. Right-click **Domain Management**, and click **Add Domain** in the local menu. In the Add Domain window (Figure 7-20) all the fields may already be filled in. If not, click browse and a list of configured domains will be presented. Select your EIM domain and click **OK**.


Add Domain - As20.itso.ibm.com

Domain information

Domain: ITSO EIM

Parent DN: None

Domain controller: as20.itso.ibm.com

Connection

☐ Use secure connection (SSL or TLS)

Port: 389

Figure 7-20 Adding your new domain to the navigation pane

Important: The options *Add domain* and *Remove domain*, available from the *Domain Management* local menu, act only locally on the client. They may be run at any time without having any effect on the operation of the EIM domain or domain controller. On the other hand the option *Delete* available from the local menu of the particular domain would delete the domain from the domain controller.

2. The added domain now appears in iSeries Navigator. Expand the domain by clicking on the **+** next to the domain name. At this point you may be prompted to specify DN and password for LDAP access. You will see two new subcategories, User Registries and Identifiers, added to the list.

   When you click User Registries there should be two user registries which the wizard has added, these were specified in step 7 on page 113. You may optionally right-click at the registries in the panel on the right and select **Properties** in the local menu. In the Properties window you may change the description set by the wizard to something more meaningful for you.

   Identifiers, which will be covered in the next section, will be the used to define the mapping of users from one system to another.

This completes adding the domain to the domain management of this client.

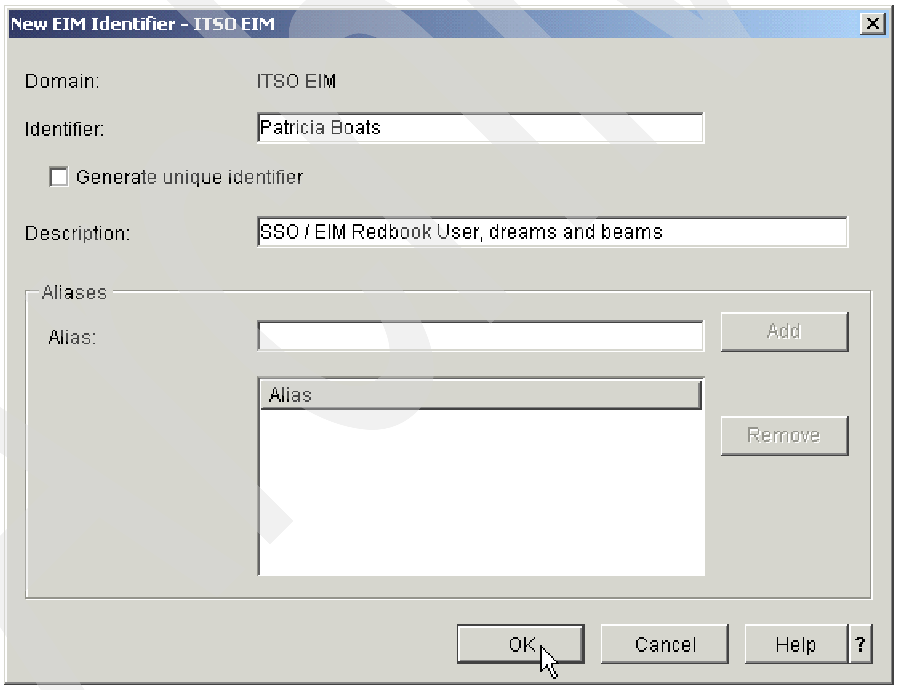
7.2.3 Using iSeries Navigator to add identifiers and associations

Prior to testing your EIM configuration you will need to create at least one user in your EIM domain. You can use iSeries Navigator to manage a small number of users in EIM. However, in iSeries Navigator you can not browse the user profiles or accounts in the user registries - you have to key in all the user IDs.

Add users to the EIM domain

Perform the following three steps to add a user (or rather an EIM ID) to the EIM domain:

1. In iSeries Navigator, expand your **iSeries -> Network -> Enterprise Identity Mapping**. If you haven't connected to the EIM domain controller previously you will be prompted to enter the distinguished name (item R from the configuration worksheet) and its password.
2. Right-click **Identifiers** in the left panel of the iSeries navigator, then select **New Identifier** in the local menu.
3. The New EIM Identifier window appears as shown in Figure 7-21. Enter the EIM Identifier of the new user you are going to add to the EIM domain. The EIM identifier has to be unique within the EIM domain. iSeries Navigator will issue an error message that indicates an identifier by this name already exists when you try to create a duplicate ID. You may chose to select the *Generate Unique Identifier* box. If you would happen to add a duplicate EIM ID, the system will append a number to the end of the ID.


New EIM Identifier - ITSO EIM

Domain: ITSO EIM

Identifier: Patricia Boats

☐ Generate unique identifier

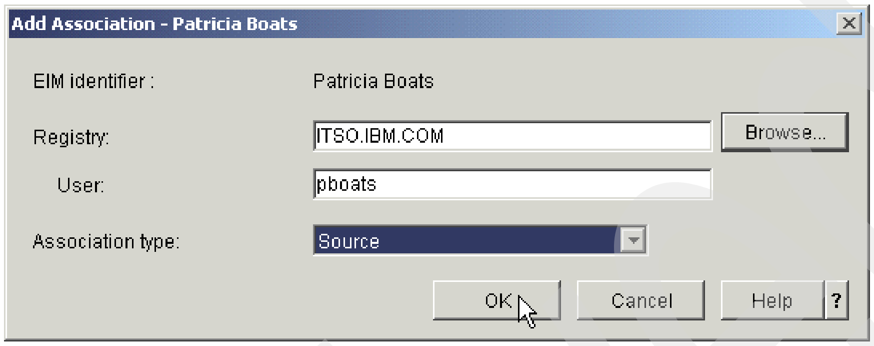
Description: SSO / EIM Redbook User, dreams and beams

Figure 7-21 Create new EIM ID

Add associations

You have added the EIM Identifier of the user in the EIM domain. Now you have to associate this identifier with the user it represents in the various registries. The following steps demonstrate how to associate the users with their identities in the user registries.

1. In the left pane under your EIM domain, click **Identifiers**. A list of identifiers will come up in the right pane of the iSeries Navigator. Right-click the user you have just added and select **Properties** from the menu.
2. The Properties window of the selected EIM ID appears. Select the **Associations** tab and click **Add**.
3. We first add the association to the Kerberos registry. In the Add Association window (Figure 7-22), click the **Browse** button for a list of the available registries.

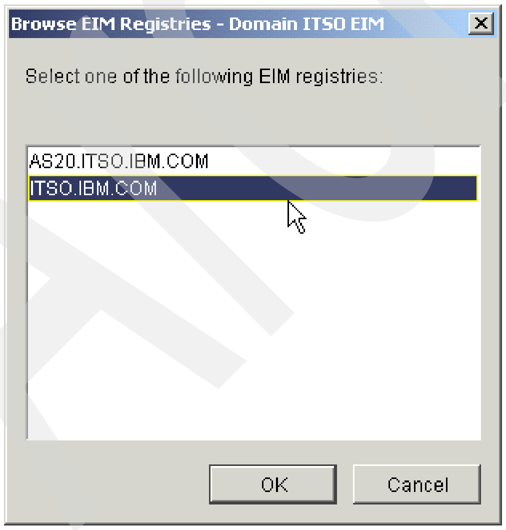


EIM identifier : Patricia Boats
Registry: ITSO.IBM.COM
User: pboats
Association type: Source

Figure 7-22 Add Kerberos association

4. The Browse EIM registries window (Figure 7-23) shows the two user registry definitions, added to the EIM domain by the EIM wizard (as you saw in Figure 7-17 on page 114):
   – Local OS/400 registry for your iSeries system, hosting the EIM domain controller, represented here by its fully qualified host name - AS20.ITSO.IBM.COM in our example.
   – Kerberos registry on the Windows 2000 server, represented here by bare domain name (that is, not representing any host name) - ITSO.IBM.COM in our example.

   Select the Kerberos registry and click **OK**.



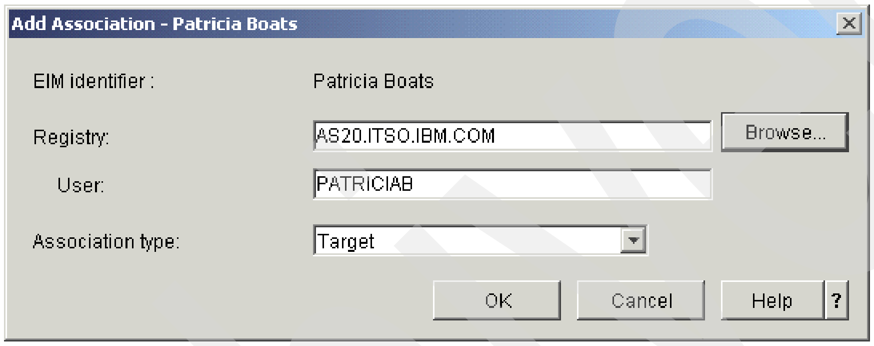
Select one of the following EIM registries:

AS20.ITSO.IBM.COM
ITSO.IBM.COM

Figure 7-23 Select Kerberos registry

5. Clicking OK takes us back to the Add Association window. Complete it as shown in Figure 7-22:
   a. Fill in the User field with the user ID which the user uses to sign on to the Windows domain - `pboats` in our example.

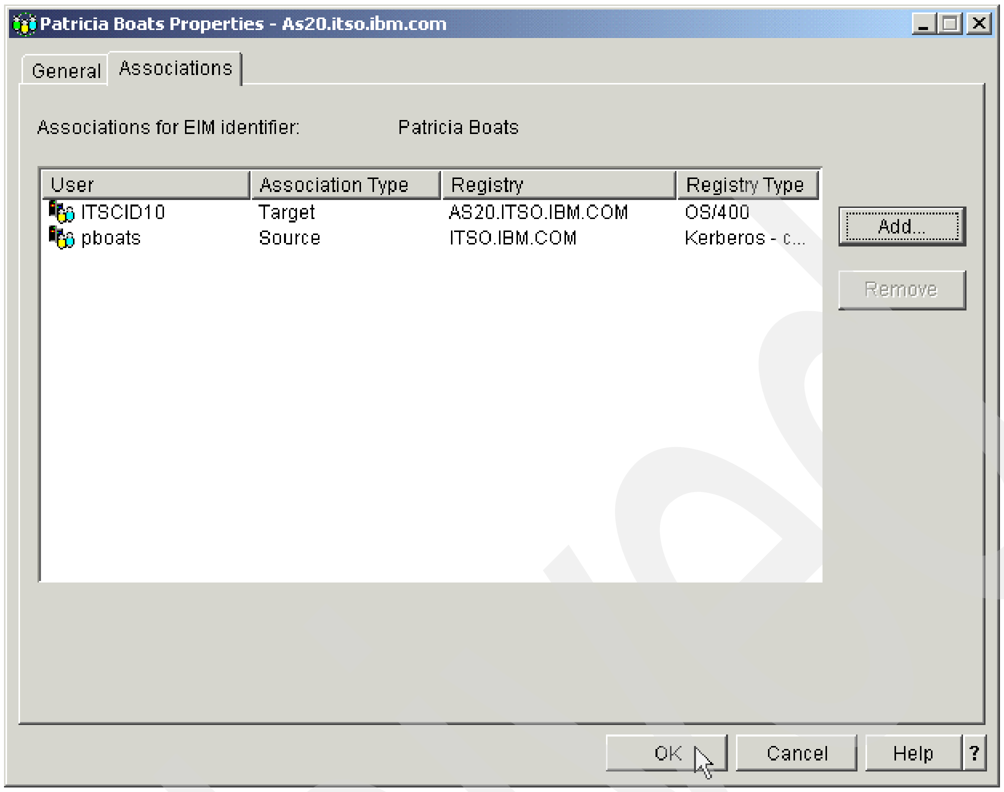
   b. Make sure that you select **Source** in the Association type combo box.
   c. Click **OK**.
6. Next you need to add the OS/400 registry:
   a. Click the **Add** button again.
   b. In the Add Association window (Figure 7-22 on page 118), click the **Browse** button.
   c. In the Browse EIM registries window select the OS/400 registry, as shown in Figure 7-23 - AS20.ITSO.IBM.COM in our case.
   d. In the Add Association window (Figure 7-24) fill in the User field with the name of the user profile of the user in OS/400.
   e. Select **Target** in the Association type combo box.
   f. Click OK in the Add Association window.



EIM identifier : Patricia Boats
Registry: AS20.ITSO.IBM.COM
User: PATRICIAB
Association type: Target

Figure 7-24 Add OS400 association

7. Now in the Properties window for the selected EIM ID (Figure 7-25), we can see both associations:
   – The OS/400 registry with the Target association - `AS20.ITSO.IBM.COM` in our example.
   – The Windows 2000 Kerberos registry with the Source association - `ITSO.IBM.COM` in our example.



Associations for EIM identifier: Patricia Boats

User	Association Type	Registry	Registry Type
ITSCID10	Target	AS20.ITSO.IBM.COM	OS/400
pboats	Source	ITSO.IBM.COM	Kerberos - c...

Figure 7-25 Both associations added

8. Verify the associations are correct and click **OK**.

This completes adding the user and his/her associations in the EIM domain. Repeat the steps for additional users.

7.3 Enable IBM iSeries applications for single signon

Now that you have completed the set-up of the components needed for the single signon, you verify the set-up by enabling two widely used IBM client applications to access the iSeries server:

- iSeries Navigator
- iSeries Access 5250 emulation

7.3.1 Getting ready

Before enabling the two applications, first ensure that these three prerequisites are met:

1. To test the EIM component of SSO you will need to have at least one user that has a user ID on the iSeries that is different than their windows logon ID. An example of the associations for such a test user was our Patricia Boats in “Add associations” on page 117. Recall that Patricia Boats logs into her Windows PC using *pboats* (source association), while her OS/400 user profile is *PATRICIAB* (target association).

Note: With a test user with same user ID in both Windows and OS/400 you would only verify correct set-up of Network Authentication Service; EIM would not be used.

2. Change the OS/400 profile of the test user to PASSWORD(*NONE) to demonstrate the full strength of this single signon solution. To do this:
   a. Sign on to OS/400 as a user with *ALLOBJ and *SECADM authorities.
   b. Enter the Change User Profile (CHGUSRPRF) OS/400 command, for example:
      ```
      CHGUSRPRF USRPRF(PATRICIAB) PASSWORD(*NONE)
      ```
3. To enable the 5250 emulation for single signon, the system value QRMTSIGN needs to be set to *VERIFY (recommended) or alternatively to *SAMEPRF (less secure environment). Do the following to check and/or change the QRMTSIGN system value:
   a. Sign on to OS/400 as a user with *ALLOBJ and *SECADM authorities.
   b. Enter the Display System Value (DSPSYSVAL) OS/400 command:
      ```
      DSPSYSVAL SYSVAL(QRMTSIGN)
      ```
   c. If the value shown is *FRCSIGNON or *REJECT, use the Change System Value (CHGSYSVAL) OS/400 command to change the setting:
      ```
      CHGSYSVAL SYSVAL(QRMTSIGN) VALUE(*VERIFY)
      ```

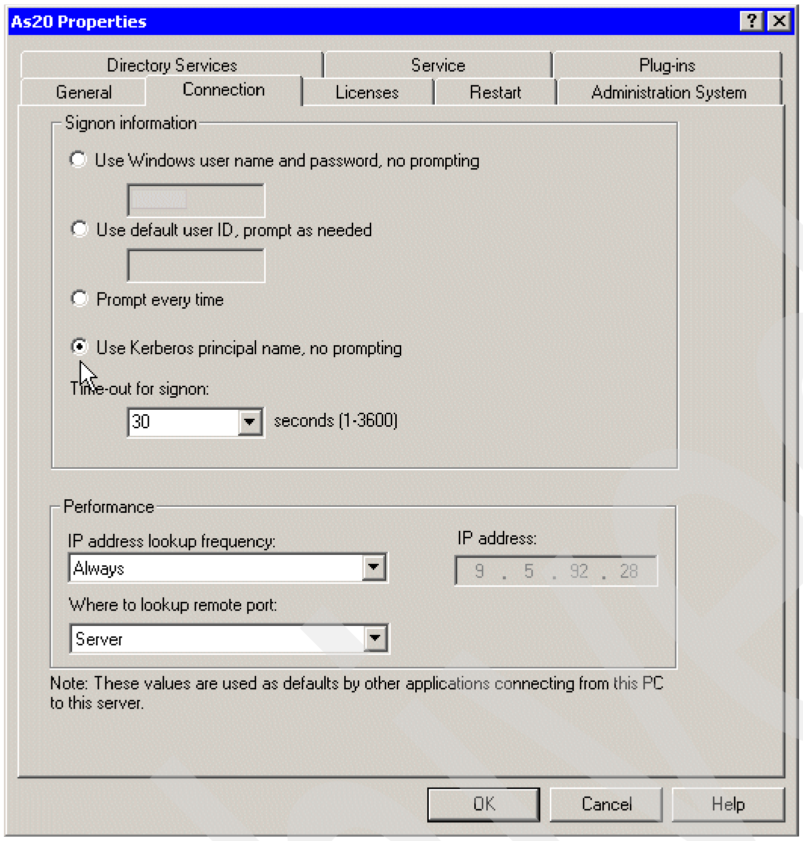
7.3.2 Enabling iSeries Navigator single signon

Perform the following steps to enable single signon for the iSeries Navigator:

1. In iSeries Navigator, right-click your iSeries server and in the local menu select **Properties**.

   If you have not been connected to the iSeries server, the connection is attempted and one of the Signon to iSeries windows may appear. You should cancel the connection attempt by clicking on the **Cancel** button or on the **No** button.
2. In the iSeries server Properties window (Figure 7-26) select the **Connection** tab. In the Signon information section, select the option **Use Kerberos principal name, no prompting**; then click **OK**.

Single signon for the iSeries Navigator is now enabled.



Signon information

( ) Use Windows user name and password, no prompting

( ) Use default user ID, prompt as needed

( ) Prompt every time

(●) Use Kerberos principal name, no prompting

Time-out for signon: 30 seconds (1-3600)

Performance

IP address lookup frequency: Always

IP address: 9 . 5 . 92 . 28

Where to lookup remote port: Server

Note: These values are used as defaults by other applications connecting from this PC to this server.

Figure 7-26 iSeries Navigator - iSeries server connection properties

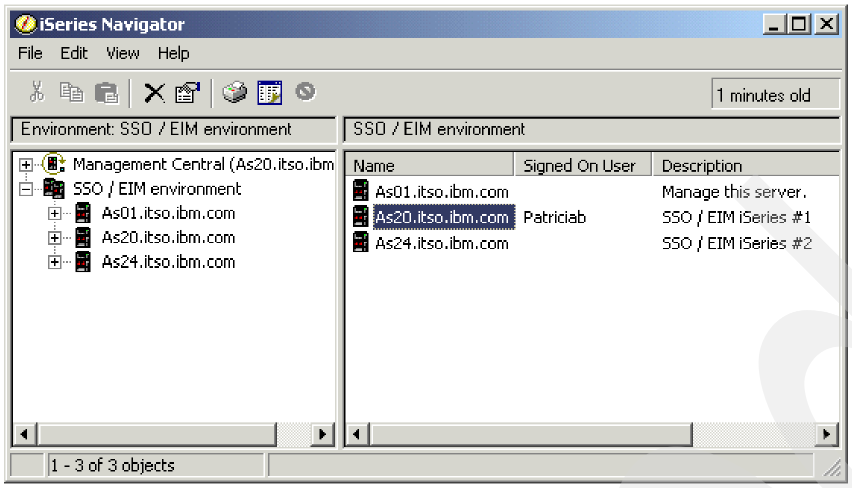
Verifying iSeries Navigator single signon

Perform the following steps to verify that single signon works for the iSeries Navigator:

1. In the left panel of the iSeries Navigator window, expand your iSeries server by clicking on the + button left to the iSeries server name.

   The connection to the iSeries server is started; note that *no signon window appeared*.

2. To find which user ID is used by the connection, click the environment in the left panel (its default name is My Connections) and press the F5 key to refresh the window contents. The user ID now appears next to the iSeries server name in the Signed On User column, as shown in Figure 7-27.



Name	Signed On User	Description
As01.itso.ibm.com		Manage this server.
As20.itso.ibm.com	Patriciab	SSO / EIM iSeries #1
As24.itso.ibm.com		SSO / EIM iSeries #2

Figure 7-27 iSeries Navigator - check signed on user

You have completed the verification of the iSeries Navigator single signon.

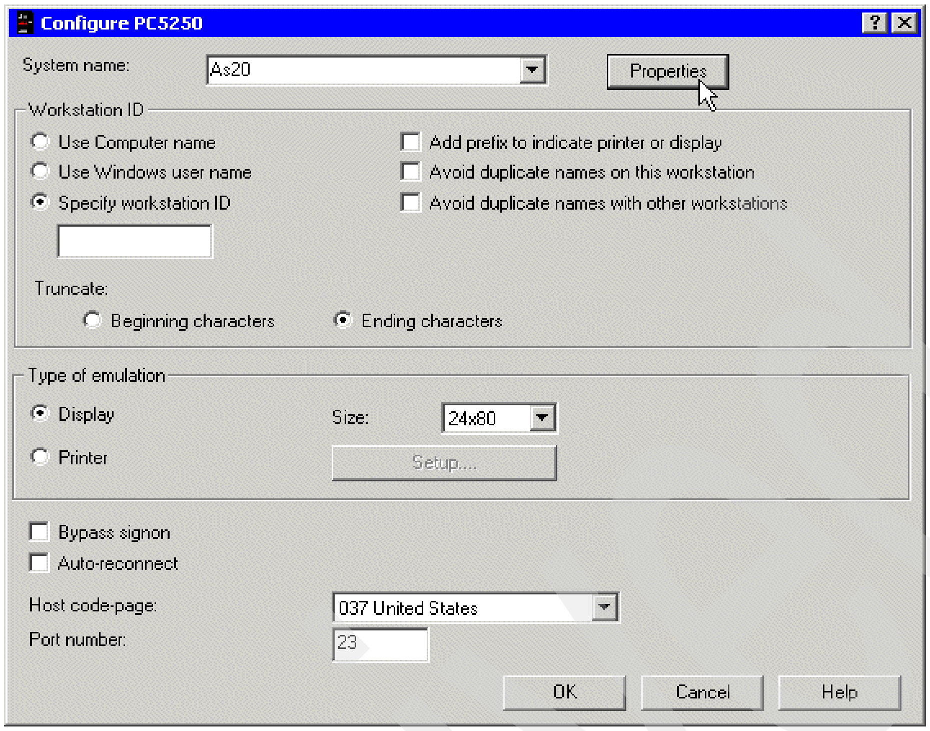
7.3.3 iSeries Access 5250 emulation single signon

In this section we demonstrate how to enable and then verify single signon for the iSeries Access 5250 emulation.

Enabling iSeries Access 5250 emulation single signon

Perform the following steps to enable single signon for the iSeries Access 5250 emulation:

1. Open the iSeries Access 5250 emulation session to your iSeries server. If the session is connected to the server, disconnect it by selecting **Communication** -> **Disconnect** in the menu.
2. Then select **Communication** -> **Configure** in the session menu.
3. In the Configure PC5250 window (Figure 7-28), click the **Properties** button.



System name: As20

Workstation ID

- ( ) Use Computer name
- ( ) Use Windows user name
- (●) Specify workstation ID
- ☐ Add prefix to indicate printer or display
- ☐ Avoid duplicate names on this workstation
- ☐ Avoid duplicate names with other workstations

Truncate: ( ) Beginning characters (●) Ending characters

Type of emulation

- (●) Display
- ( ) Printer

Size: 24x80

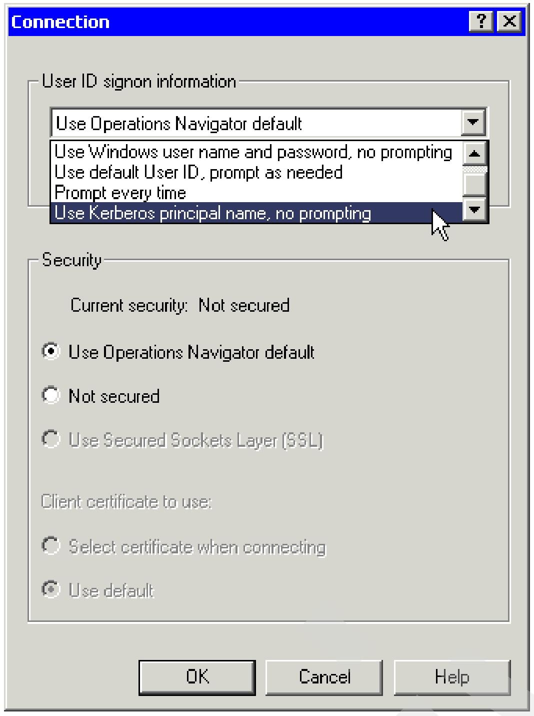
- ☐ Bypass signon
- ☐ Auto-reconnect

Host code-page: 037 United States

Port number: 23

Figure 7-28 TN5250 - select Properties

4. When the Connection window (Figure 7-29) comes up, go to the combo box in the section labeled User ID signon information. Scroll the combo box to the very bottom and select the option **Use Kerberos principal name, no prompting**, as shown in the Figure 7-29. Alternatively, you can select the option **Use Operation navigator default**. This option allows you to control all connections to a particular iSeries server from one place. You have previously selected the Kerberos-type connection for this iSeries server in the iSeries Navigator connection properties (step 2 on page 121). Click **OK**.
5. To save the session set-up, select **File** -> **Save** from the 5250 session window menu. Single signon for the iSeries Access 5250 emulation is now enabled.


Connection

User ID signon information

Use Operations Navigator default

Use Windows user name and password, no prompting
Use default User ID, prompt as needed
Prompt every time
Use Kerberos principal name, no prompting

Security

Current security: Not secured

(●) Use Operations Navigator default
( ) Not secured
( ) Use Secured Sockets Layer (SSL)

Client certificate to use:

( ) Select certificate when connecting
(●) Use default

Figure 7-29 TN5250 - select Use Kerberos principal name

Verifying iSeries Access 5250 emulation single signon

Perform the following steps to verify that single signon works for the iSeries Access 5250 emulation:

1. Select **Communication** → **Connect** from the menu of the 5250 session window.
2. The session connects to the iSeries server and you are signed on; note that *no signon window appeared*.
3. To find out which user profile the session is using issue the OS/400 command Display Workstation User (DSPWSUSR), no parameters are required. The results will look similar to Figure 7-30.

```
 Display Work Station User                   AS20
                                                             12/02/03  15:30:55
 User . . . . . . . . . . . . . . . . . . . :   PATRICIAB
   Text . . . . . . . . . . . . . . . . . . :     Patricia Boats, EIM Residency
862644897
 Work station . . . . . . . . . . . . . . . :   P782XNXKA
   Text . . . . . . . . . . . . . . . . . . :     Device created for AS20.


 Number of interactive jobs in session  . . :   1
 Interactive job currently active . . . . . :   A
   Interactive job A  . . . . . . . . . . . :     073320/PATRICIAB/P782XNXKA
   Interactive job B  . . . . . . . . . . . :     *NONE
```

Figure 7-30 DSPWSUSR results

You have completed the verification of the iSeries Access 5250 emulation single signon.

8

Other scenarios

We described our basic scenario in Chapter 3, “The redbook example scenario” on page 29 and its basic implementation in Chapter 7, “Enabling Network Authentication Service and Enterprise Identity Mapping” on page 97.

In this chapter we describe these additional scenarios and their implementations:

- “The Bike Shop scenario” on page 128
- “Using remote SQL with single signon” on page 143
- “Enabling another iSeries server for single signon” on page 145
- “Enabling NetServer for single signon” on page 153
- “Enabling Domino Web Access for single signon and EIM” on page 162
- “Enabling Web Express Logon for WebSphere Host on-Demand” on page 179

8.1 The Bike Shop scenario

Our company, The Bike Shop, is growing. In the past remote sales people, managers, and directors could dial into the company network or use a virtual private network (VPN) to access pricing information. However, telecommunication costs have skyrocketed as the company has expanded; network dial-in is slow; and users find the legacy interface cumbersome. The Bike Shop has considered replacing the application and moving it to its Web application server. The costs though are too high.

The company has chosen to integrate and extend its Enterprise Information System (EIS) too, through its J2EE application server. There are many challenges, especially considering security. The application server has a set of unique users, and so does the iSeries that serves the EIS application. The EIS iSeries has complex object level and database authorities. How does The Bike Shop keep existing users and authorities and still extend their application securely? EIM.

8.1.1 EIM solution overview

Users needing pricing will request it from the Internet using a secure connection. The application server will authenticate the users, and forward user name, password and requests to the EIS system. The EIS system will authenticate again. Once authenticated on the EIS system, the user is then mapped to a user that has authority to run the price look-up application. The EIS application will get the handle of the mapped user and will run as the mapped user.

Depending on the authority level of the mapped user, the EIS will return pricing. For regular sales people, EIS will return retail pricing only. For sales mangers, EIS will return both retail and wholesale pricing. For directors, EIS will return retail, wholesale, and actual cost. Users that are not authorized will receive no results.

8.1.2 The components

This section discusses all of the components, both physical and logical, that are part of this scenario. Figure 8-1 shows the components in The Bike Shop's network. There is a brief description of each of the components that will come into play for this scenario.

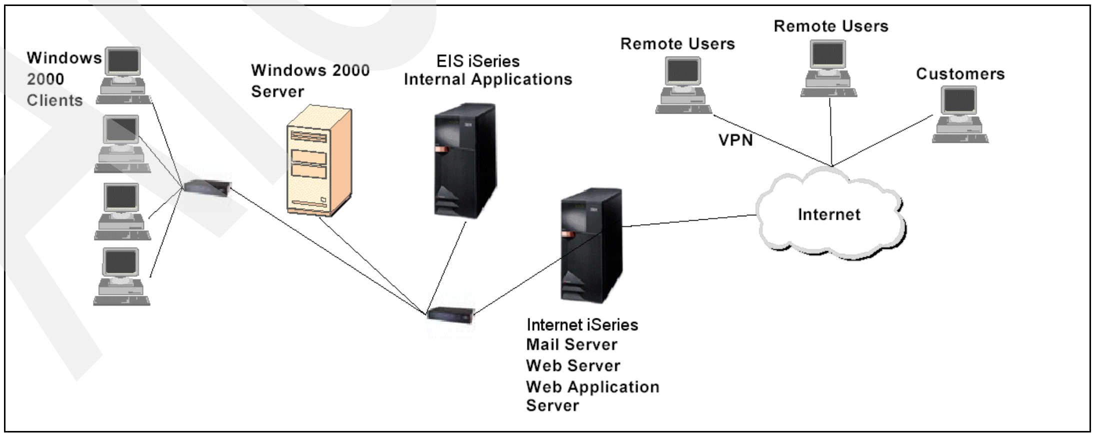


Figure 8-1 The Bike Shop's network layout

The Internet iSeries

The Internet iSeries hosts a variety of applications that are accessible from the Internet. Most importantly, it hosts our Web application server that serves our J2EE applications. These applications include bikePrice.jsp and BPServlet. bikePrice.jsp is a simple presentation layer interface to BPServlet. BPServlet forces basic authentication and connects to the EIS iSeries using the SocketConnection class. The source for these programs is available at the download site for this redbook. These Java applications are discussed in 8.1.3, “The J2EE application in more detail” on page 130.

The EIS iSeries

The EIS iSeries hosts the internal applications that will be extended to the Web. It also hosts the EIM domain that we use to map users.

The EIS application

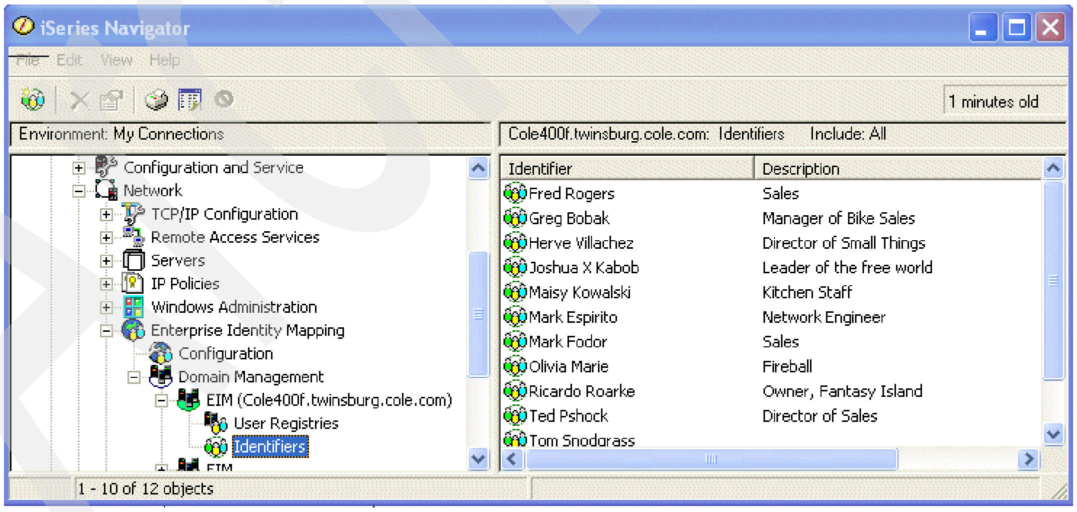
The EIS hosts an enterprise suite of applications for managing a bike shop. For the purpose of our scenario, we only are extending the price information application. The full source is available from the download site for this redbook:

```
ftp://www.redbooks.ibm.com/redbooks/SG246975
```

The RPGLE program, BPPRICESQL, executes SQL statements over logical files that have unique authority settings. The program has no awareness of who is running the program. Authority is determined at the object level for each logical file. Consequently only the calling program and the operating system know which user is trying to access the data.

The EIM domain

The EIM domain for the whole enterprise is served on the EIS iSeries. From this domain we will get the source and target identity for each unique entity accessing our system. Figure 8-2 shows a partial list of the identifiers in the EIM domain. Only users with the correct source and target associations will be able to access both the Web application and the EIS application.



Cole400f.twinsburg.cole.com: Identifiers   Include: All

Identifier	Description
Fred Rogers	Sales
Greg Bobak	Manager of Bike Sales
Herve Villachez	Director of Small Things
Joshua X Kabob	Leader of the free world
Maisy Kowalski	Kitchen Staff
Mark Espirito	Network Engineer
Mark Fodor	Sales
Olivia Marie	Fireball
Ricardo Roarke	Owner, Fantasy Island
Ted Pshock	Director of Sales
Tom Snodgrass	

Figure 8-2 The EIM Domain

The EIM application

The EIM application running on the EIS iSeries is an RPGLE program (GETEIMMAP) that calls the C APIs discussed in 9.7, “C EIM API” on page 221.

The source is available from the download site for this redbook:

ftp://www.redbooks.ibm.com/redbooks/SG246975

Once a source identity is successfully mapped, the program gets the handle of the mapped identity, and then invokes the EIS application. GETEIMMAP is discussed in more detail in “GETEIMMAP RPGLE” on page 135.

8.1.3 The J2EE application in more detail

In this section we take a closer look at the Java portion of our application. Please note that these are for example purposes only. They could be much more robust and secure.

bikePrice.jsp

The bikePrice.jsp is a very simple Java Server Page (JSP) that prompts a user for a model number of a bike. The entire JSP is in Example 8-1. It really is quite a simple JSP. You can see that it provides basic presentation and posts a part number to BPServlet, and nothing else.

Example 8-1 bikePrice.jsp

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"%>

<META http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<META name="GENERATOR" content="IBM WebSphere Studio">
<TITLE>partPrice.jsp</TITLE>
</HEAD>
<BODY>
  <H1>It's A Bike Shop Bike Prices</H1>
  <P>This application allows you to retrieve pricing information for our extensive line of
bicycles.
     <br>You must be a sales person, manager, or director to use this system.
     <br>Please have your username and password handy.
  </P>
  <H2>Enter your 6 character model number:</H2>
  <FORM action="BPServlet" method="post">
    <INPUT type="text" name="partnum"/>
    <INPUT type="submit" value="Submit" />
  </FORM>
</BODY>
</HTML>
```

BPServlet.java

BPServlet is the heart of the Java portion of our example. After receiving a part number from bikePrice.jsp, it forces basic authentication and then connects to the EIS. The doAction() method provides a good outline of our flow.

Example 8-2 doAction method of BPServlet

```
/**
* Logic for doPost and doGet
*/
public void doAction(HttpServletRequest req, HttpServletResponse resp)
   throws ServletException, IOException {
```

```
   PrintWriter out = resp.getWriter();
   boolean authorized = false;
   String auth = req.getHeader("Authorization");
   if (auth != null){
        authorized = getAuth(req, resp, out, auth);
   } else {
        assertAuth(resp, out);
   }

   if (!authorized)
        assertAuth(resp, out)

} //doAction
```

The doAction() method checks for an authorization header. If it is present, it calls the getAuth() method to authenticate and authorize the request. If there is no authorization header, the doAction() calls assertAuth() to force basic authentication.

The assertAuth() method

The assertAuth() method (Example 8-3) forces basic authentication.

Example 8-3 assertAuth() method forces basic authentication

```
private void assertAuth(HttpServletResponse resp, PrintWriter out){

   resp.setHeader("Www-Authenticate", "Basic realm=\"Put Your Realm Here\"" );
   resp.setContentType("text/html; charset=UTF-8");
   resp.setStatus(HttpServletResponse.SC_UNAUTHORIZED);
   out.println("<html>");
   out.println("<head>");
   out.println("<title>401 Authentication Required</title>");
   out.println("</head>");
   out.println("<body>");
   out.println("<h1>401 Authentication Required</h1>");
   out.println("<p>Some message about not being able to verify authorization to access the 
requested resource.</p>");

// Only use the following if you have some additional message
   out.println("<p>Some additional message with details on the authorization 
problem.</p>");
   out.println("<hr>");
   out.println("<address>Some address or information on who to contact about 
problems.</address>");
   out.println("</body>");
   out.println("</html>");

   }//end assertAuth
```

Setting the response status to SC_UNAUTHORIZED forces the browser to prompt for user name and password. In Figure 8-3 we see the dialog box presented by the Web browser. The remaining HTML is only displayed if the user fails to authenticate.

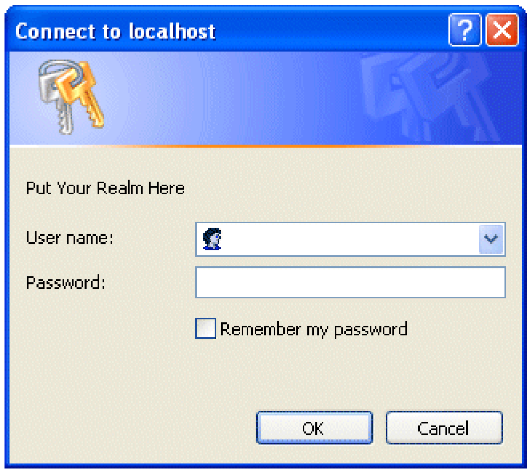


Figure 8-3 Basic authentication dialogue in Microsoft Internet Explorer

The getAuth() method

The getAuth() method (Example 8-4) parses the username and password from the authorization header and does an LDAP bind to authenticate.

Example 8-4 The getAuth() method

```
private boolean getAuth(HttpServletRequest req, HttpServletResponse resp,PrintWriter out,
String auth)
{

/**
* Replace these values with your ldap server values
**/
   String ldapURL = "ldap://myldaphost.com:389/";
   String cn      = "cn=";
   String ldapDN  = ",cn=users,dc=myldaphost,dc=com";

   String partnum = null;
   partnum = req.getParameter("partnum");


   boolean authorized = false;
   System.out.println("auth = " + auth);
   int i = auth.indexOf(" ");
      String type = auth.substring(0, i);
      if (type.equalsIgnoreCase("Basic"))
      {
      String user = null;
      String password = null;
      String creds = auth.substring(i + 1);
          if (creds != null)
          {
         password = "";
         try{
          user = new String(new sun.misc.BASE64Decoder().decodeBuffer(creds));
         } catch(IOException iox){
            iox.printStackTrace();
            return authorized;
         }
            int j = user.indexOf(':');
            if (j != -1)
            {
            password = user.substring(j + 1);
            user = user.substring(0, j);
            }//if j
       } //if creds
// Do your processing here
           Context ldapCtx = null;
           Binder  ldapBnd = new Binder();

           ldapCtx = ldapBnd.bind(ldapURL, cn+user+ldapDN, password);

           if (ldapCtx != null){
         showPrices(partnum, user, password, resp,out);
         ldapBnd.closeContext(ldapCtx);
            authorized = true;
           }
   } //if Basic

      return authorized;

}//end getAuth
```

There are some very important points to note about the getAuth() method. First, you may use what ever authentication method you want. In this example we use basic authentication which transmits a BASE 64 encoded username and password combination. Second, be sure to use secure transport (HTTPS) when using basic authentication because BASE 64 is easily decoded by standard Java APIs. Third, this example uses LDAP bind as the authentication point. Although you may use any validation method available, we chose to base our application on LDAP authentication because the applications on both Internet iSeries and the EIS iSeries systems can use the user provided ID and password to authenticate the user. This avoids the necessity of the iSeries Internet system having to know the user's corresponding user profile and password on the EIS iSeries system.

After authentication is complete a connection is made to the EIS. The connection is made using itso.eim.socket.SocketConnection. It is a simple socket connection that sends user, password, and part number to a socket listener on the EIS. You may use a number of methods to connect to the EIS. The SocketConnection class is among the simplest but not the most secure. Also, there may be problems with correlating responses under load. In a production environment, it may be better to use a J2EE Connection Architecture (JCA) application provided by your EIS vendor. Another less complicated alternative is to use Java Message Service (JMS) if you have a message queue product like WebSphere MQ available.

8.1.4 The EIS applications

The EIS applications are RPGLE programs that provide the following functions:

1. Listen for requests from WebSphere Application Server
2. Authenticate the user sent from WebSphere Application Server
3. Map the user to the target user
4. Assume the identity of the target user
5. Call the legacy application as the target user
6. Return a response to WebSphere Application Server

The code for these applications is available at the download site for this redbook.

SOCKETSVR RPGLE

SOCKETSVR RPGLE is the main controller and interface to the EIS from WebSphere Application Server. It is a simple socket listener that calls C APIs to listen for requests and reply to WebSphere Application Server. It also takes on the function of translating from ASCII to EBCDIC and back. Most importantly, it performs the function of switching handles to run the legacy application as the target user.

Let's step through core SOCKETSVR functionality.

1. After receiving a message, it must be translated from ASCII to EBCDIC so that it can be read by the EIS. The call to *translate* is actually a call to *QDCXLATE* from the system libraries (Example 8-5).

Example 8-5 Translate from ascii to ebcdic

```
*
*  translate from ascii to ebcdic...
*
c                   eval      xlate = bytesrd
c                   callp     translate(xlate:bufferds:ebcdic)
c                   exsr      writelog
c                   exsr      getdata
```

2. SOCKETSVR calls getprofile to authenticate the user received from WebSphere Application Server (Example 8-6). This is a second level check so our application doesn't automatically trust the user sent over by WebSphere Application Server. Calls to *getprofile* and *rlsprofile* are actually calls to *QSYGETPH* and *QSYRLSPH* respectively. The user profile is then released so that the application can assume the mapped user identity.

Example 8-6 Second level authentication

```
*  Authenticate user profile and password...
*
c                   callp     getprofile(userid:password:handle:errds)
 *
c                   if        handle <> *loval
*  Release user profile...
c                   callp     rlsprofile(handle)
```

3. Map to target user (Example 8-7). This calls GETEIMMAP which is discussed in greater detail in "GETEIMMAP RPGLE" on page 135.

Example 8-7 Mapping to target user

```
*  EIM Mapping...
c                   callp     geteimmap(userid:eimuser:errmsg)
*
```

4. Now we authenticate the target user on the local system. We do this so that we can get the user's handle from the operating system to run our legacy application as the target user. The call to *getprofile* (Example 8-8) is actually a call to *QSYGETPH*. Note that the password is *NOPWD meaning that the user does not have a password on this system.

Example 8-8 Verifying local user

```
*  Authenticate user profile and password...
c                   callp     getprofile(eimuser:'*NOPWD':handle:errds)
```

5. After getting the target user's handle, we can run the legacy application as that user (Example 8-9). The program running as the target user is now effectively that user and subject to all the security constraints of that user.

Example 8-9 Call the legacy application

```
*
c                   callp     socketsql(partnum:rtndata)
*
```

6. Before returning results to the WebSphere Application Server application, *translate* is called again to convert EBCDIC to ASCII (Example 8-10). The results are written to a buffer that sends the results back to the WebSphere Application Server application.

Example 8-10 Translate back to ASCII

```
*
c                   eval      bufferds = %trim(rtndata)+crlf
c                   eval      xlate    = %len(%trim(bufferds))
c                   callp     Translate(100:bufferds:ascii)
*
c                   eval      byteswr = write(socketcli:
c                                           %addr(buffer):
c                                         %size(bufferds))
```

7. Finally, the target user handle is dropped, and replaced by the original user the program was called under (Example 8-11). This is very important. Only user profiles with *ALLOBJ authority and *SECADM authority are authorized to the QSYGETPH program to get handles for user profiles without being required to pass the user profile.

Example 8-11 The user ID is restored to the original requesting user ID

```
*  Release EIM user profile...
c                   callp     rlsprofile(handle)
c                   endif
 * Set profile back to master
c                   callp     setprofile(master:errds)
* Close Socket...
```

GETEIMMAP RPGLE

GETEIMMAP is the fundamental EIM process for mapping a source user to a target user. Here are the 4 steps to the process and their C API calls:

1. Create Handle -- you must create controlling handle as base for EIM session. This handle is used through out the program for all EIM API calls:

   `int eimCreateHandle(EimHandle * eim, char * ldapURL,EimRC * eimrc)`

2. Connect to EIM Domain -- This step is needed to establish connection to correct domain to map source identity to EIM identifier then to target:

   `int eimConnect(EimHandle * eim, EimConnectInfo connectInfo, EimRC * eimrc)`

3. Get target identity -- this is where the source identity is mapped to a source identity for authorization to EIS program:

   ```
   int eimGetTargetFromSource(EimHandle * eim,
   char * sourceRegistryName,
   char * sourceRegistryUserName, char * targetRegistryName,
   ```

```
char * additionalInformation, unsigned int lengthOfListData, EimList * listData,
EimRC * eimrc)
```

4. End EIM session. This just normal close of EIM session:

```
int eimDestroyHandle(EimHandle * eim, EimRC * eimrc)
```

Each one of these is part of the RPGLE service program QSYEIM in the QSYS library. Let’s take a closer look at the API calls in our RPGLE program.

eimCreateHandle

Example 8-12 shows what the call looks like in RPGLE.

Example 8-12 Call to eimCreateHandle API

```
c                   eval      rtcd  =
c                              eimCreateHandle(handle:*NULL:eimrc)
```

The variables handle and eimrc are data structures. They are defined in the EIM copybook found in QRPGLESRC source physical file in the QSYSINC library. They are not explicitly declared as pointers because data structures in RPGLE are really pointers already. Note that for the ldapURL we pass *NULL. This tells the program to use whatever domain controller the operating system is configured to use. You may pass a fully qualified LDAP URL, but be aware that you will need *SECADM authority for it to work.

If `rtcd` is equal to 0 after the call, you have a valid handle and will be able to connect to your EIM domain. The error handling API is discussed in “eimErr2String” on page 138.

eimConnect

This is the API that makes a connection to your EIM domain. Before you can make the call, you must populate the data structure for *EimConnectInfo* (Example 8-13).You can use any authentication method accepted by the LDAP server hosting the EIM domain. We have chosen the basic distinguished name and password method for this scenario. Whatever method you use to authenticate to the EIM domain controller, you must authenticate as a use that has at least EIM lookup authority within the domain.

Example 8-13 Call to eimConnect API

```
c                   eval      BindDn = %trim(BindDn)+null
c                   eval      BindPw = %trim(BindPw)+null
c                   eval      eimtype    = *zero
c                   eval      eimotect00 = *zero
c                   eval      eimotect   = *zero
c                   eval      eimbd00    = %addr(BindDn)
c                   eval      eimbp00    = %addr(BindPw)
c                   eval      rtcd  =
c                              eimConnect(handle:eimci:eimrc)
```

Before passing user name and password, we strip away any trailing white space and terminate our values with null. We do this because C expects null to terminate the value. Without null, any garbage in memory could be appended to your values and you could not authenticate on the EIM domain. We pass *zero for the remaining values telling EIM that we are doing a simple bind and that we are passing the password as clear text. If `rtcd` is 0, we can get our target user.

eimGetTargetFromSource

The eimGetTargetFromSource API is heart and soul of EIM. This is where EIM will map a user ID from one system (source -- the user's LDAP name in our example) to another user ID on another system (target -- the user's OS/400 user profile in our example). At this point we know who our source user is and which registry it is from. We have to tell EIM to get us the target based on that user. Example 8-14 demonstrates the call to the API.

Example 8-14 Call the eimGetTargetFromSource API

```
c                   eval      srcRegName = %trim(srcRegName)+null
c                   eval      srcUsrName = %trim(srcUsrName)+null
c                   eval      trgRegName = %trim(trgRegName)+null
c                   eval      rtcd  =
c                              eimGetTrgFrmSrc(handle:
c                                          srcRegName@:
c                                          srcUsrName@:
c                                          trgRegName@:
c                                          additional@:
c                                          listSize:
c                                          eimlist@:
c                                          eimrc)
```

Again, we terminate our string values with null to avoid unintended garbage from being attached to them. After our call, if rtcd equals 0, the data structure eimlist@ points to should be populated. We have to parse the list to get our target user profile (Example 8-15). Because it is possible that a single person may have more than one user ID in the target registry, the API must be able to return multiple associated IDs; hence we need to parse the results to make sure only one ID was returned. If your application encounters more than one returned target identity and has no way of knowing which one to choose, the application should return an error message stating something to the effect that this user has multiple IDs on this system and it doesn't know which one to pick.

Important: This potential situation bears more discussion. Oftentimes certain people will have multiple IDs on one system. For example. system administrators may have an ID they use when performing administrative tasks and a different one using corporate applications as a "normal" user. The additional@ data structure was intended to provide a way to allow applications to avoid this issue or to deal with it more effectively. Additional information can be added to a target association and the application can tell EIM to return only target associations that also include this additional information. Even if more than one target ID association exists, EIM will only return the one or ones that have that information. Alternatively, an application could choose to loop through each returned target ID and explicitly check for the additional information. Normally you would just let EIM do it. For purposes of clarity and brevity, our example ignores this issue.

Example 8-15 Parsing the eimlist

```
c                   eval      Entry@  = %addr(listData)
c                   eval      RtnUser = %subst(enentry:1:enlength)
```

All of the difficult work is in the setup of the data structures. listData is the last 32 bytes of eimlist. Entry@ is a based pointer to a data structure that will hold the values at the address of listData. That data structure tells us the position and length of our target user.

eimDestroyHandle

To finish, we close our connections and destroy our handle (Example 8-16). The call is very similar to eimCreateHandle.

Example 8-16 Call to eimDestroyHandle API

```
c                   eval      rtcd  =
c                              eimDestHandle(handle:eimrc)
```

eimErr2String

Up until this point, we have not explained eimrc in our examples. It is the reason code returned if there is an error. To get a meaningful value from the code, use eimErr2String (Example 8-17).

Example 8-17 Returning error codes

```
c                   eval      errMsg@ = eimErr2String(eimrc)
c                   eval      RtnMsg  = %str(errMsg@:%size(errMsg))
```

BPRICESQL RPGLE

This is our legacy program. It has no dependencies on any of the programs for sockets or EIM. It can be run directly from a command line or from a menu of The Bike Shop's enterprise package. The values returned to the end user are determined by the authorities set on the files the program accesses. That is what makes EIM special. With no changes to our legacy code, and no authority changes on our system, an end user can run this code and never know that identity mapping has occurred.

8.1.5 Notes about setting up and compiling the example code

This section provides some tips for setting up the WebSphere Application Server to EIS sample code. Some general guidelines and assumptions:

- Have Project Development Manager (PDM) installed and know how to compile DDS and RPGLE files.
- Create a library named EIMLIB to run the RPGLE examples without modifications.
- Create source physical files (SRC-PF) for QDDSSRC and QRPGLESRC.
- Know how to use WebSphere Studio Application Developer for the Java programs.

The code examples, named WebSphereScenarios.zip, can be downloaded here:

ftp://www.redbooks.ibm.com/redbooks/SG246975

After downloading and extracting the code, FTP all iSeries code to your test iSeries like using the syntax shown in Example 8-18.

Example 8-18 FTP DDS and RPGLE files to iSeries

```
C:\>ftp to.your.iSeries
Connected to to.your.iSeries
220-QTCP at WWW.YOURiSeries.COM.
220 Connection will close if idle more than 5 minutes.
User: bobakgp
331 Enter password.
Password:
230 BOBAKGP logged on.
ftp> cd eimlib
```

```
250 "EIMLIB" is current library.
ftp> lcd eimrb\rpgle
Local directory now C:\EIMRB\RPGLE.
ftp> verbose
Verbose mode Off .
ftp> mput q*
mput QDDSSRC.BPRICEL1? y
mput QDDSSRC.BPRICEL2? y
mput QDDSSRC.BPRICEL3? y
mput QDDSSRC.BPRICEPF? y
mput QRPGLESRC.BPRICESQL? y
mput QRPGLESRC.EIM? y
mput QRPGLESRC.GETEIMMAP? y
mput QRPGLESRC.SOCKETCPY? y
mput QRPGLESRC.SOCKETSRV? y
ftp>
```

Note: Do not switch to binary using `bin`. Use the default of ascii so that the code is translated from ASCII to EBCDIC automatically. Also, you must create source physical files before uploading the code to your iSeries. QDDSSRC has a record length of 92, and QRPGLESRC must have a record length of 112.

The Java examples are discussed in "Compiling and deploying the Java examples" on page 141.

8.1.6 Compiling files and setting up the physical file and logical file authorities

There are four steps to this process:

1. Change the source type of each file.
2. Compile the DDS files.
3. Set the correct authorities.
4. Populate the BPRICEPF.

Change the source type of each file

The source type of each file is lost in the FTP process; you have to set it manually.

1. Start PDM (STRPDM) and navigate to the sample code.
2. For each file, take option 13.
3. Set source type for BPRICEPF to "PF".
4. Set source type for the remaining files to "LF".

Compiling the files

Compile the files in PDM. You must compile BPRICEPF first. After it has compiled, you may compile the logical files in any order.

Set the correct authorities

There is one physical file (BPRICEPF) stores pricing data, and three logical files (BPRICEL1, BPRICEL2, and BPRICEL3) that describe the view of that data. For this example, all users/groups that have access to the logical files must have *USE authority to BPRICEPF. Be sure you have enough authority to add data to this file. Our example program is only aware of the logical files, and cannot access the physical file directly.

Set up the authorities on the logical files this way:

- BPRICEL1 should only allow directors *USE, and *PUBLIC should be set to EXCLUDE.
- BPRICEL2 should only allow managers *USE, and *PUBLIC should be set to EXCLUDE.
- BPRICEL3 should only allow sales *USE, and *PUBLIC should be set to EXCLUDE.

Populate BPRICEPF

You need to populate BPRICEPF to return prices. You may do this using any utilities you have available, but Example 8-19 uses the iSeries SQL interface (`STRSQL`).

Example 8-19 Populate BPRICEPF using SQL

```
===> insert into EIMLIB/SOCKETPF (frpart, frret, frwhs, frcst)
     values('000003', 000120, 000090, 000075)
```

Do this as many times as you would like, creating unique part numbers each time.

8.1.7 Compiling the RPGLE examples

After uploading the examples:

1. Set the source type on each file.
2. Edit system specific settings in GETEIMMAP.
3. Add library QSYSINC to your library list.
4. Create binding directory for QSYEIM service program.
5. Compile the programs.

Set the source type on each file

1. Start programing development manager (STRPDM) and navigate to the sample code.
2. For each file, take option 13.
3. Set source type for BPRICESQL to “RPGLESQL”.
4. Set source type for the remaining files to “RPGLE”.

Edit system specific settings in GETEIMMAP

GETEIMMAP has system specific values for LDAP distinguished name and password, and source and target registries. Change the following lines to the values that apply to your environment.

```
d BindDn         s            80    inz('cn=administrator')
d BindPw         s            80    inz('ldappw')
d srcRegName     s            80    inz('COLE400A.TWINSBURG.COLE.COM')
d trgRegName     s            80    inz('COLE400F.TWINSBURG.COLE.COM')
```

You may want to modify the programs to get these values from a properties file. It is not recommended that you store user names and passwords in programs. Passwords should not normally be stored in plain text.

Add library QSYSINC to your library list

You need QSYSINC in your library list so that the EIM copybook will be included in your code. The EIM copybook provides all of the prototypes for the service program QSYEIM which is the interface to the C APIs for EIM.

Create binding directory for QSYEIM service program

In order to use the C APIs for EIM, the GETEIMMAP program is bound to QSYEIM. So that you don't have to explicitly create and bind modules to the program at compile time, GETEIMMAP has an "h" specification to bind to objects in a named binding directory.

```
h bnddir('QSYEIMBD') dftactgrp(*NO)
```

From the iSeries command line, create the binding directory and add QSYSEIM as an entry like this:

```
CRTBNDDIR BNDDIR(EIMLIB/QSYEIMBD)
```

And:

```
ADDBNDDIRE BNDDIR(EIMLIB/QSYEIMBD) OBJ((QSYS/QSYEIM))
```

You will only have to create the binding directory once.

Compile the RPGLE programs

You can compile all of the RPGLE programs in PDM normally. Do not compile SOCKETCPY. SOCKETCPY is a copy book for the SOCKETSVR program.

If everything compiles correctly, you are now ready to run the EIS code. Submit SOCKETSVR, and it will listen on port 50007 for requests from WebSphere Application Server.

8.1.8 Compiling and deploying the Java examples

Like the RPGLE examples, you will have to edit the Java code to get it to run correctly in your environment. For convenience, all of the Java code is included in EIMWeb.war. This is easily imported into WebSphere Studio Application Developer (WSAD) to edit and rebuild. Stand alone code is also available for download, however we will focus on importing the WAR file into WebSphere Studio Application Developer.

There are three steps for this procedure:

1. Import EIMWeb.war into WebSphere Studio Application Developer.
2. Edit Java source and build project.
3. Deploy code.

Import EIMWeb.war into WebSphere Studio Application Developer

These are the steps for importing EIMWeb.war into WebSphere Studio Application Developer:

1. Create a new Enterprise Application project in WebSphere Studio Application Developer.
2. Import Web Application project into WebSphere Studio Application Developer.

Edit Java source and build project

The BPServlet contains specific references to the LDAP server for the simple bind. In WebSphere Studio Application Developer, edit the getAuth() method of BLServlet.java and change the code in Example 8-20.

Example 8-20   LDAP bind parameters

```
/**
* Replace these values with your ldap server values
**/
String ldapURL = "ldap://host.name.com:389/";
String cn      = "cn=";
String ldapDN  = ",cn=users,dc=host,dc=name,dc=com";
```

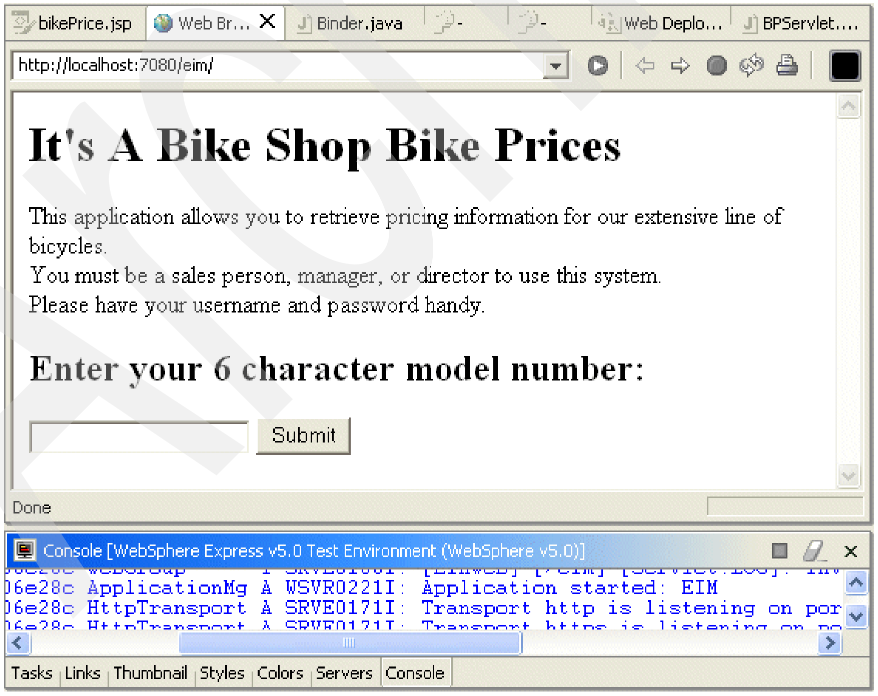
Before you can compile this servlet and build the project, you must add binder.jar to your build path. Use the online help in WebSphere Studio Application Developer for instructions. Click **Project -> Rebuild Project** to compile the code.

Deploy code

You may test this code on the built in WebSphere Test Environment (WTE) that is part of WebSphere Studio Application Developer, or you may deploy it on a WebSphere Application Server test instance in your environment. You should probably test it in the WTE before deploying it on a server.

To run in WTE, right-click your project and select Run on server. Select an existing server configuration, or a new one. This project will run in both Versions 4 and 5 of WebSphere Application Server. You must edit the server's configuration and add binder.jar to the classpath.

To run on a WebSphere Application Server, export the enterprise project containing EIMWeb.war to a Enterprise Application Archive (EAR). Save your EAR on your server and use WebSphere Application Server Admin Console to import your application. After importing your EAR, copy binder.jar into WEB-INF/lib of your application install directory. Start and launch your application. Figure 8-4 shows how a successful launch looks.



It's A Bike Shop Bike Prices

This application allows you to retrieve pricing information for our extensive line of bicycles.
You must be a sales person, manager, or director to use this system.
Please have your username and password handy.

Enter your 6 character model number:

```
)6e28c ApplicationMg A WSVR0221I: Application started: EIM
)6e28c HttpTransport A SRVE0171I: Transport http is listening on por
```

Figure 8-4   Successfully deployed in WTE

8.2 Using remote SQL with single signon

When you enabled single signon with iSeries Navigator you also enabled remote SQL (See 7.3.2, “Enabling iSeries Navigator single signon” on page 121). The following section is a short demonstration on how to verify that it is actually working. We are going to load data into an Excel table from a sample table QCUSTCDT, which comes with the OS/400 in the library QIWS.

Perform the following steps:

1. If you are signed on to your Windows client, sign off, to make sure that you don’t have an existing connection to your iSeries system.
2. As a test, sign on to your client as a user that has different user IDs for Windows and for your iSeries system - otherwise you would only verify the Network Authentication Service, not the EIM.
3. Start Microsoft Excel. From the Excel main Menu (Figure 8-5), select **Data -> Transfer Data from iSeries**.

> **Note:** The commands **Transfer Data From iSeries** and **Transfer Data To iSeries** are plug-ins installed with the iSeries navigator. So is the Client Access tool bar (the only tool bar shown in Figure 8-5, in the upper left corner). You can use icons in this tool bar instead the **Transfer Data To/From iSeries** menu commands.

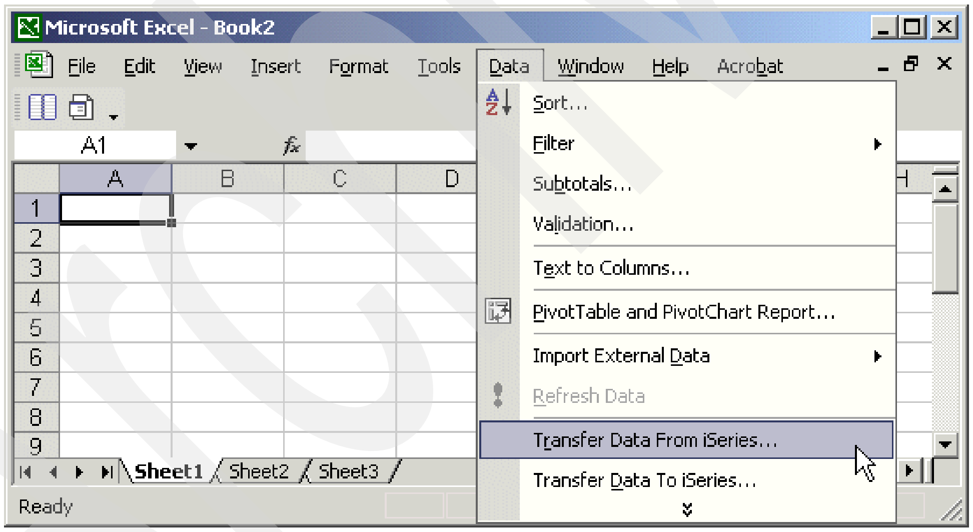
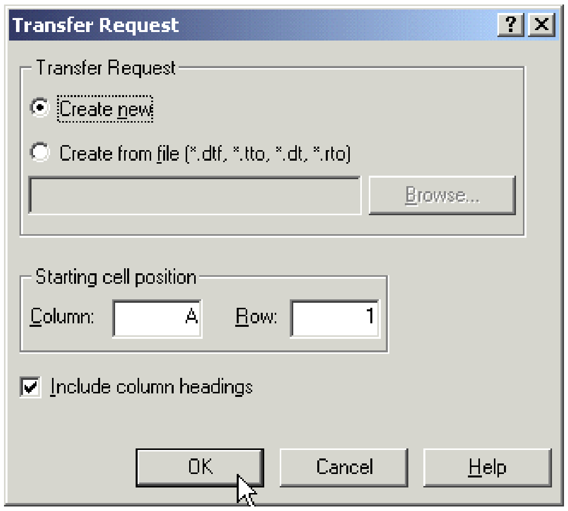


Figure 8-5 Excel menu - Data - Transfer Data from iSeries

4. In the Transfer Request window (Figure 8-6), check the **Create New** check box, and then click **OK**.


Transfer Request

Transfer Request

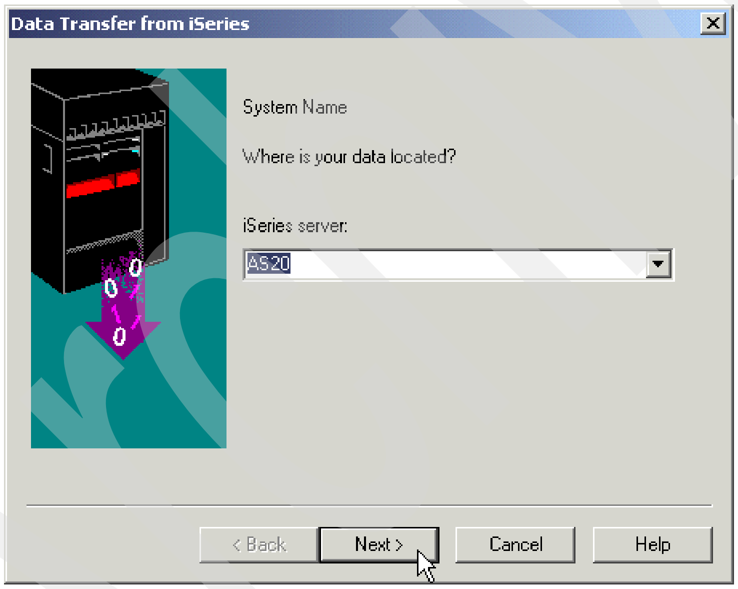
(● Create new) ○ Create from file (*.dtf, *.tto, *.dt, *.rto)

Starting cell position
Column: A Row: 1

☑ Include column headings

Figure 8-6 Transfer Request - Create new

5. In the first Data Transfer window (Figure 8-7), select your iSeries in the combo box, and then click **Next**.


Data Transfer from iSeries

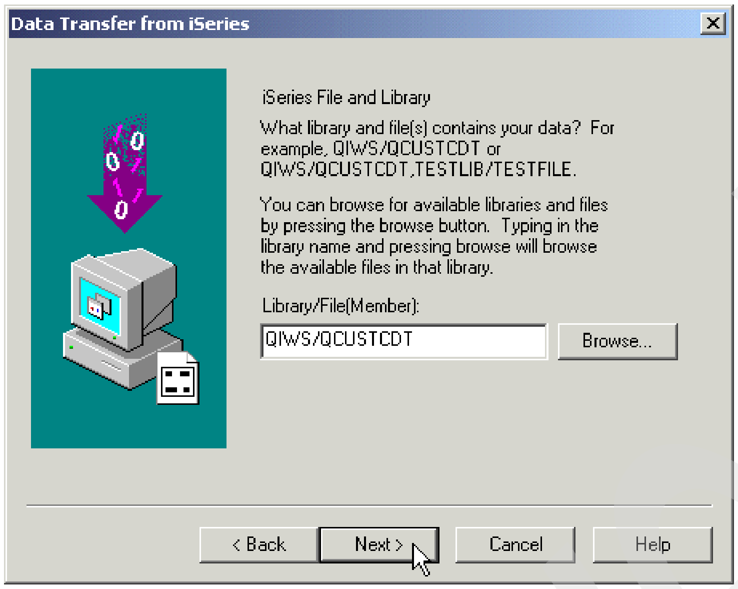
System Name

Where is your data located?

iSeries server: AS20

Figure 8-7 Data Transfer from iSeries - select iSeries server

6. In the Data Transfer window (Figure 8-8) enter `QIWS/QCUSTCD` in the Library/File(Member) field; then click Next.


Data Transfer from iSeries

iSeries File and Library

What library and file(s) contains your data? For example, QIWS/QCUSTCDT or QIWS/QCUSTCDT,TESTLIB/TESTFILE.

You can browse for available libraries and files by pressing the browse button. Typing in the library name and pressing browse will browse the available files in that library.

Library/File(Member):

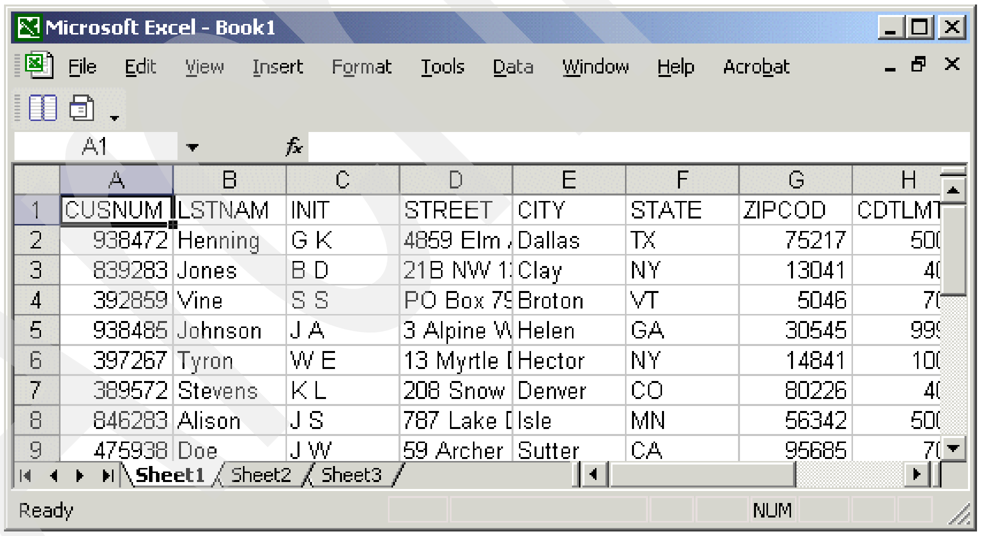
QIWS/QCUSTCDT

Figure 8-8 Data Transfer from iSeries - specify `library/file`

7. As you need not specify any transfer options or save the transfer request, just click **Next** and then **Finish** in the next two Data Transfer windows.

   No signon window appears and the Excel table is filled in with data from your iSeries (Figure 8-9).

   The verification of the remote SQL usage is completed.


Microsoft Excel - Book1

	A	B	C	D	E	F	G	H
1	CUSNUM	LSTNAM	INIT	STREET	CITY	STATE	ZIPCOD	CDTLMT
2	938472	Henning	G K	4859 Elm	Dallas	TX	75217	50
3	839283	Jones	B D	21B NW 1	Clay	NY	13041	4
4	392859	Vine	S S	PO Box 79	Broton	VT	5046	7
5	938485	Johnson	J A	3 Alpine W	Helen	GA	30545	99
6	397267	Tyron	W E	13 Myrtle	Hector	NY	14841	10
7	389572	Stevens	K L	208 Snow	Denver	CO	80226	4
8	846283	Alison	J S	787 Lake	Isle	MN	56342	50
9	475938	Doe	J W	59 Archer	Sutter	CA	95685	7

Figure 8-9 Data downloaded to Excel table

8.3 Enabling another iSeries server for single signon

To enable another iSeries server for single signon, you need to do the following tasks:

- Configure Network Authentication Service for the new iSeries server.
- Add the new iSeries server to the EIM domain.
- Add associations to the EIM IDs of the users.

8.3.1 Before you begin

Before you start enabling another iSeries server, you have to:

- Check that you have met all the prerequisites on this iSeries server.
- Check that you have collected all the information you will need to enter during the set-up.

You can refer back to your configuration worksheets that you created from the blank worksheets in Table D-1 on page 240 for checking prerequisites and collecting the configuration information.

8.3.2 Configuring the Network Authentication Service

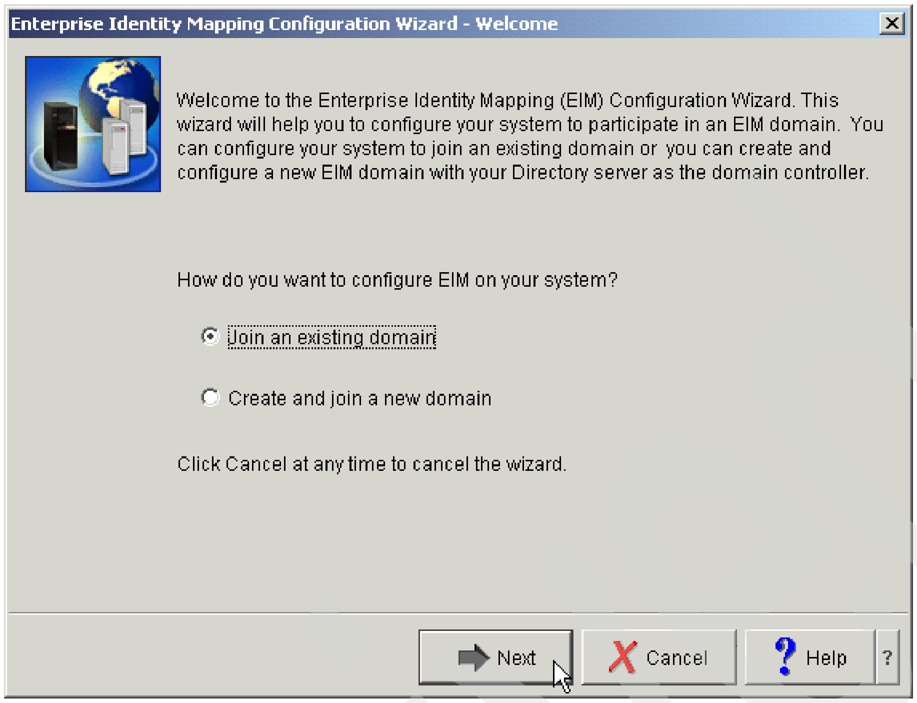
You configure the Network Authentication Service exactly the same way you did it with your first iSeries server, as described in 7.1, “Configure Network Authentication Service” on page 98 after completing the configuration worksheet from Appendix D-2, “Planning worksheet” on page 241.

8.3.3 Adding the iSeries server to the EIM domain

Most of the steps needed to add the new iSeries server to the EIM domain are similar to those performed when enabling your first iSeries server. For this reason we concentrate more on the steps which differ and for the similar steps we refer back to 7.2.1, “Using EIM configuration wizard” on page 111.

To add the new iSeries server to the EIM domain perform the following steps:

1. In iSeries Navigator, expand the new iSeries server you are enabling. Then expand **Network -> Enterprise Identity Mapping** and right-click **Configuration** and then select **configure** (or **reconfigure** if you have already attempted to configure EIM on this system before).
2. In the EIM wizard Welcome window (Figure 8-10) select the option Join an existing domain; then click **Next**.


Enterprise Identity Mapping Configuration Wizard - Welcome

Welcome to the Enterprise Identity Mapping (EIM) Configuration Wizard. This wizard will help you to configure your system to participate in an EIM domain. You can configure your system to join an existing domain or you can create and configure a new EIM domain with your Directory server as the domain controller.

How do you want to configure EIM on your system?

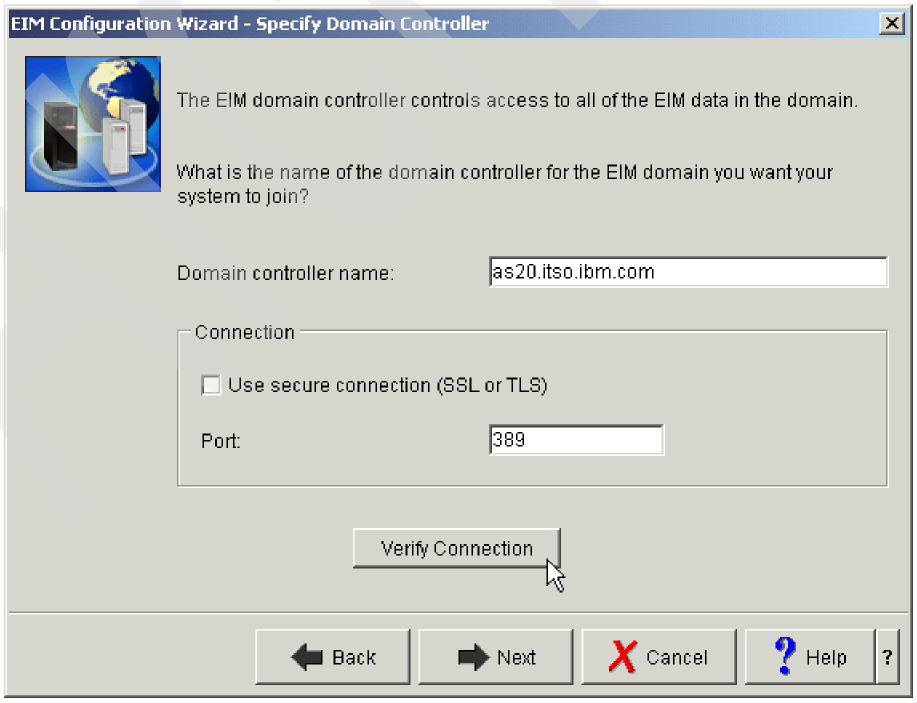
[✓ Join an existing domain]

Create and join a new domain

Click Cancel at any time to cancel the wizard.

Figure 8-10 Join an existing domain

3. Specify Domain Controller window appears (Figure 8-11). Enter the fully qualified host name of your iSeries server hosting your EIM domain controller in the Domain controller name field.
4. Click **Verify** to verify the connection (refer to Step 6 on page 113 if you need more details). Then click **Next** in the Specify Domain Controller window.


EIM Configuration Wizard - Specify Domain Controller

The EIM domain controller controls access to all of the EIM data in the domain.

What is the name of the domain controller for the EIM domain you want your system to join?

Domain controller name: as20.itso.ibm.com

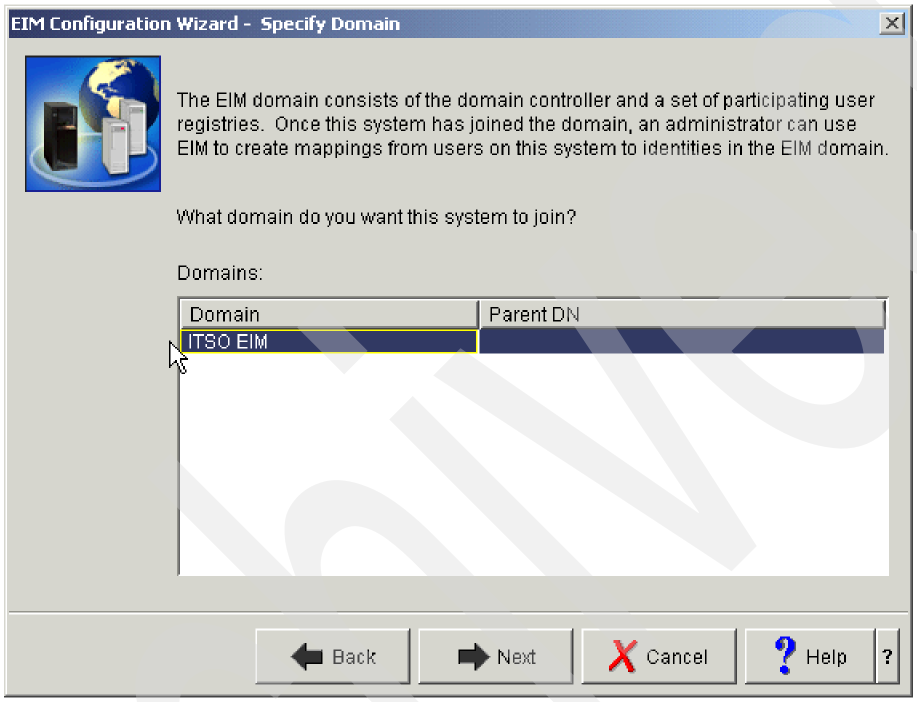
Connection

Use secure connection (SSL or TLS)

Port: 389

Figure 8-11 Specify Domain Controller

5. Specify the user and password to be used by the wizard in the Specify User For Connection window (Figure 7-16 on page 113). We recommend that you create a new LDAP user and add it to the EIM Admin group. Use this user ID and password for the system. Creating LDAP users is outside the scope of this redbook so we will use the LDAP administrator's ID and password instead.
6. Click the **Verify** button to verify the connection (refer to step 6 on page 113 if you need more details). Then click **Next** in the Specify User For Connection window.
7. In the Specify Domain window (Figure 8-12) select the domain and click **Next**.


EIM Configuration Wizard - Specify Domain

The EIM domain consists of the domain controller and a set of participating user registries. Once this system has joined the domain, an administrator can use EIM to create mappings from users on this system to identities in the EIM domain.

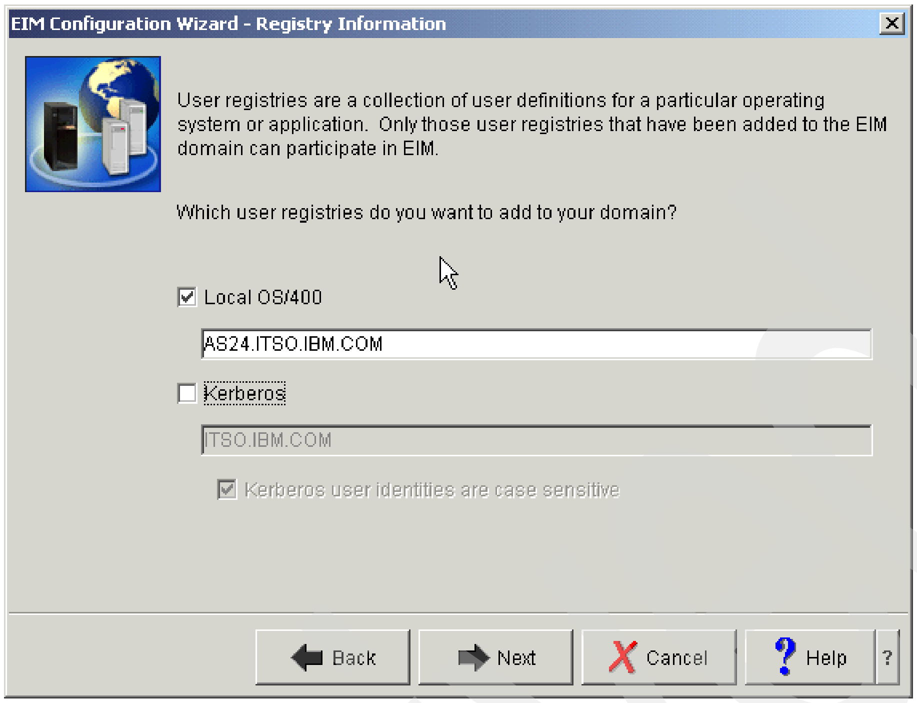
What domain do you want this system to join?

Domains:

Domain	Parent DN
ITSO EIM	

Figure 8-12 Specify domain

8. In the Registry Information window you add only the local OS/400 registry this time because the Kerberos registry was already configured with the first system. Deselect the click box for Kerberos and click **Next**.


EIM Configuration Wizard - Registry Information

User registries are a collection of user definitions for a particular operating system or application. Only those user registries that have been added to the EIM domain can participate in EIM.

Which user registries do you want to add to your domain?

☑ Local OS/400

AS24.ITSO.IBM.COM

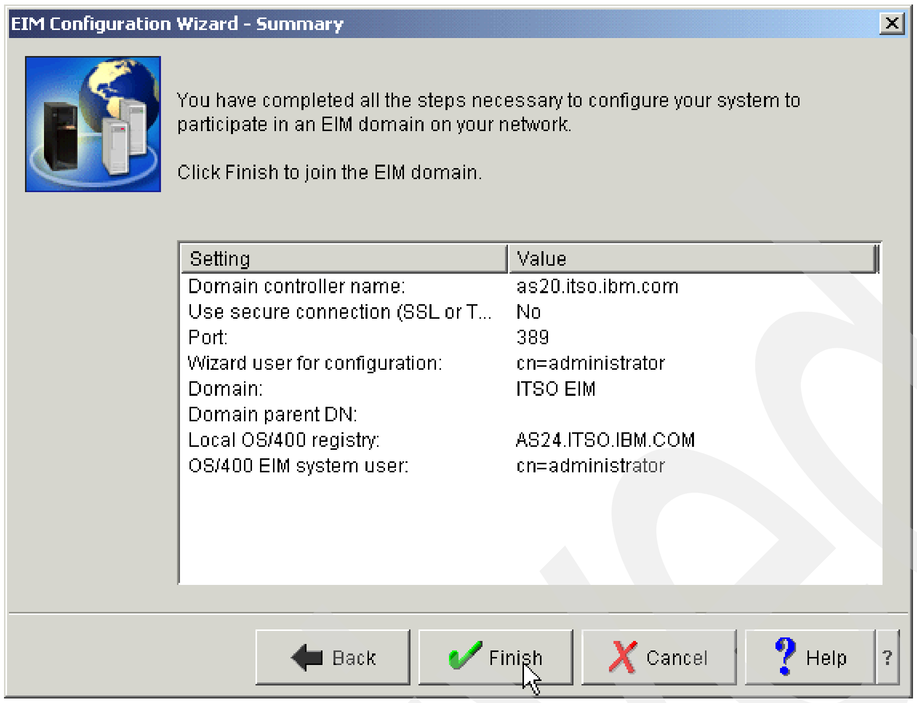
☐ Kerberos

ITSO.IBM.COM

☑ Kerberos user identities are case sensitive

Figure 8-13 Add only local OS400 registry

9. In the Specify EIM System User window you can again use item R from the configuration worksheet. This account will be used to connect to the EIM domain controller by various operating system functions.
10. Again, click the **Verify** button to verify the connection (refer to step 6 on page 113 if you need more details). Then click **Next** in the Specify EIM System User window.
11. Review the settings (Figure 8-14) you have specified. If they are correct, click **Finish**, or if you need to make changes, select the **Back** button to change your entries.



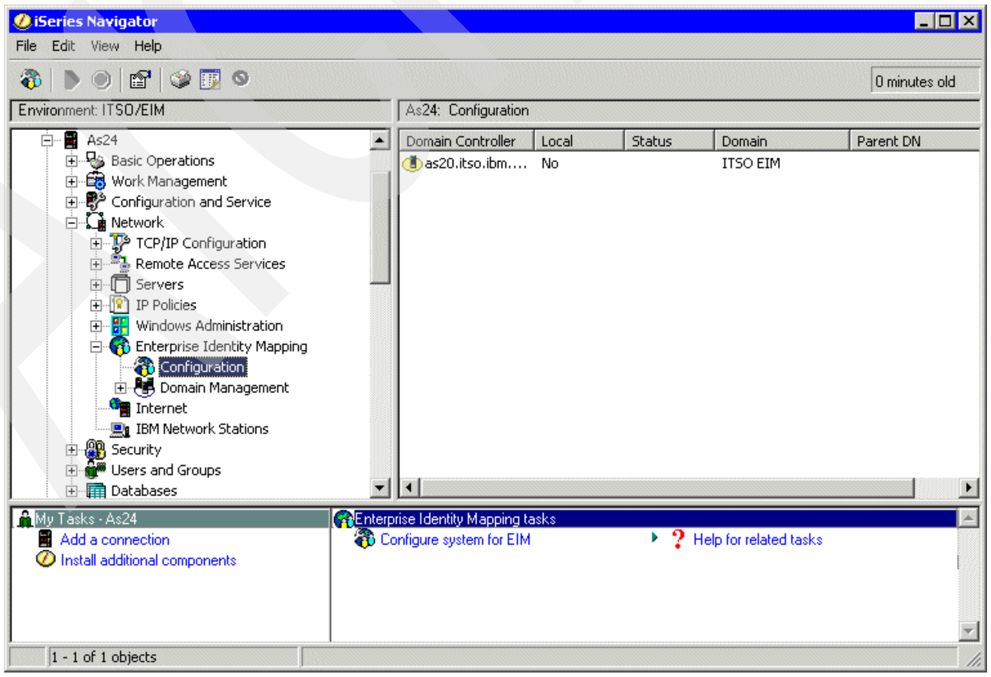
You have completed all the steps necessary to configure your system to participate in an EIM domain on your network.

Click Finish to join the EIM domain.

Setting	Value
Domain controller name:	as20.itso.ibm.com
Use secure connection (SSL or T...	No
Port:	389
Wizard user for configuration:	cn=administrator
Domain:	ITSO EIM
Domain parent DN:	
Local OS/400 registry:	AS24.ITSO.IBM.COM
OS/400 EIM system user:	cn=administrator

Figure 8-14 EIM configuration wizard summary

12. In the Configuration In Progress window you can observe the wizard as it is performing necessary steps to make your iSeries server join the EIM domain. Wait till the wizard completes.
13. In the left panel of the iSeries Navigator select **Configuration** under **Enterprise Identity Mapping**. In the right panel you see that your new iSeries server joined domain (Figure 8-15).

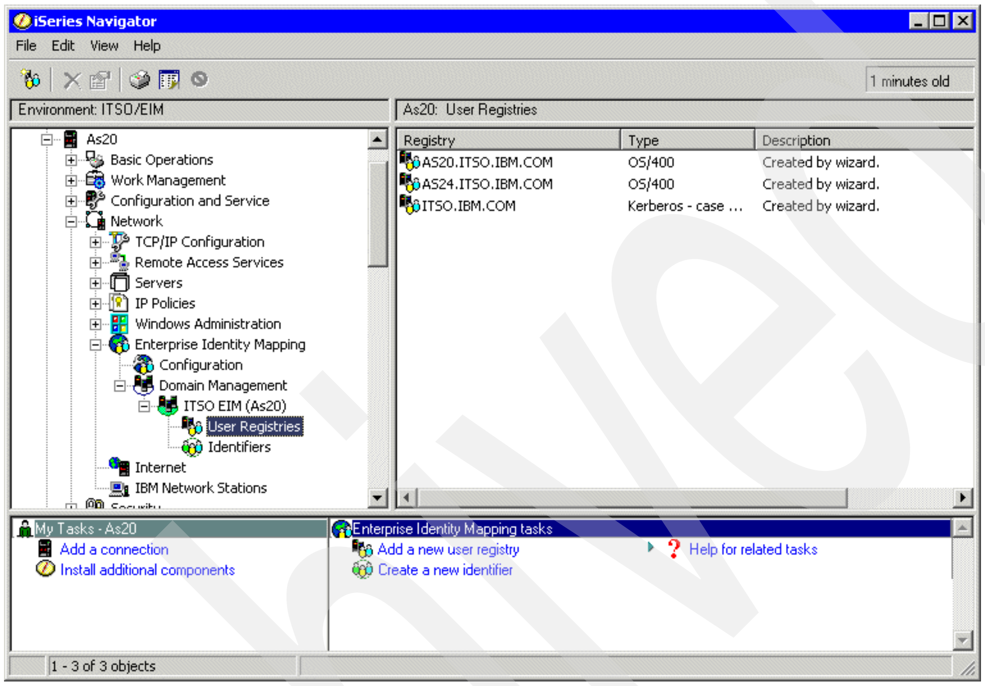


Domain Controller	Local	Status	Domain	Parent DN
as20.itso.ibm....	No		ITSO EIM	

Figure 8-15 iSeries server joined the domain

14. In the left panel of the iSeries Navigator, now move to your other iSeries server, hosting the EIM domain controller. Select **Enterprise Identity Mapping -> Domain Management -> User Registries**. If you haven't connected to the EIM domain controller previously, you will be asked to connect now. In the panel on the right you see the registry of your new iSeries server just added to the EIM domain (Figure 8-16).

This completes adding your new iSeries server to your EIM domain.



Registry	Type	Description
AS20.ITSO.IBM.COM	OS/400	Created by wizard.
AS24.ITSO.IBM.COM	OS/400	Created by wizard.
ITSO.IBM.COM	Kerberos - case ...	Created by wizard.

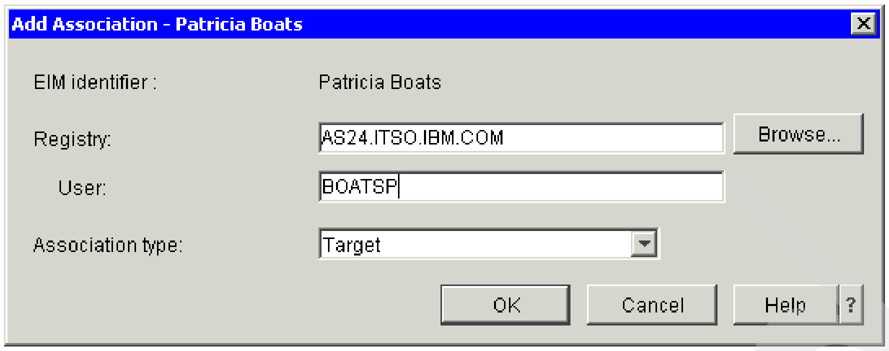
Figure 8-16 User registry added

8.3.4 Adding associations

To enable the users to use single signon on your new iSeries server, you need to add associations to their EIM IDs. We shall demonstrate what is needed in iSeries Navigator.

To add the needed associations to an EIM ID perform the following steps in iSeries Navigator:

1. In the iSeries Navigator left panel, expand your iSeries **server -> Network -> Enterprise Identity Mapping** -> **Identifiers**. In the panel on the right, right-click the EIM ID you want to add the associations to and then select **Properties**.
2. When the Properties window opens select the **Associations** tab and click **Add**.
3. In the Add Associations window (Figure 8-17), click the **Browse** button and select your new Series server in the Browse EIM registries window, and then click **OK**. After you return to the Add Association window select **Target** in the Association type combo box, click **OK**.



EIM identifier : Patricia Boats

Registry: AS24.ITSO.IBM.COM

User: BOATSP

Association type: Target

Figure 8-17 Add target association

The properties window should now display the new association.

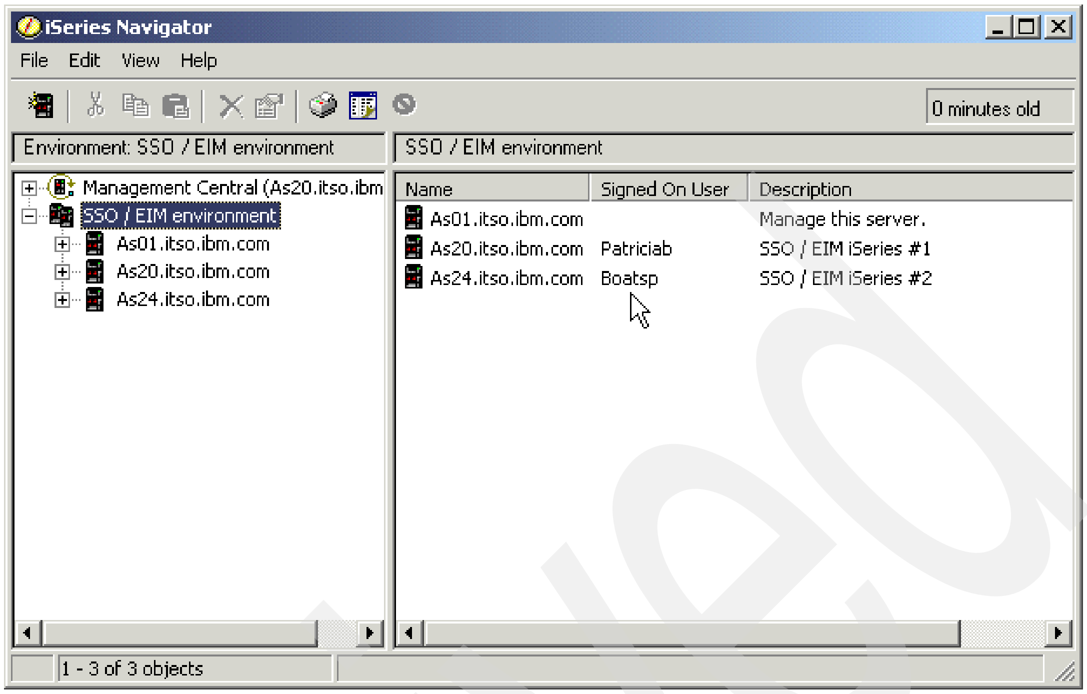
8.3.5 Verify single signon for your new iSeries server

To test our scenario with Patricia we created yet another user ID on the second iSeries. She now has three different IDs, one for Windows, one for AS20, and a new unique ID for AS24. We set her password to `PASSWORD(*NONE)` on both of the iSeries servers to demonstrate the full strength of this SSO solution.

Note: You would not have to create unique IDs on each of the systems. The mapping still takes place but having different user profiles better illustrates that the mapping occurred.

For the verification perform the following steps:

1. Sign on to your Windows client with a user ID of a test user just described.
2. In iSeries Navigator expand both your iSeries servers. Note that no signon window appeared.
3. In the left panel of the iSeries navigator select the environment, containing the two iSeries servers, as shown in Figure 8-18. Press F5 for refresh.
4. In the panel on the right look at the user ID of the signed on user on both your iSeries servers (Figure 8-18). Note that the user IDs are different and that they also differ from the user ID with which you sign on to your Windows client.



Name	Signed On User	Description
As01.itso.ibm.com		Manage this server.
As20.itso.ibm.com	Patriciab	SSO / EIM iSeries #1
As24.itso.ibm.com	Boatsp	SSO / EIM iSeries #2

Figure 8-18 Signed on users

5. Optionally, you can also verify the single signon for the 5250 emulation to your new iSeries server.

This ends the verification procedure for adding another iSeries to your EIM domain.

8.4 Enabling NetServer for single signon

To enable NetServer for single signon, follow these steps:

- Check prerequisites and collect the configuration information.
- Prepare parallel use of single signon and legacy connection to NetServer.
- Check and set up NetServer properties.
- Create three Kerberos principals in Microsoft Active Directory KDC.
- Create key table entries for these principals on your iSeries server.
- Verify the single signon to the NetServer.

Tip: If you are still working with a basic Network Authentication Service configuration it would be far less steps to simply run the Network Authentication wizard again. In Step 6 of 7.1.1, “Setting up Network Authentication Service with iSeries Navigator wizard” on page 99 (Figure 7-4 on page 103) put a check in the iSeries NetServer check box and let the wizard create the keytab entries for you.

8.4.1 Getting ready

Since you have already verified most of the prerequisites required when you started your EIM and Kerberos configuration using Table D-1 on page 240 there is only a few steps to complete before enabling NetServer for single signon. However, NetServer is going to be the exception to the rule when it comes to phased implementation with EIM. We are able to configure NetServer so it will allow connections from both a Windows 2000/XP domain and an NT domain (using earlier Windows clients). This was not originally possible with OS/400 R520 but APAR MA27136 was written to allow this environment although there is one caveat. You must have all of your 2000/XP users mapped through EIM before you configure NetServer to accept Kerberos tickets.

In other words the EIM identifier must have a source association with the Kerberos registry, and a target association with the OS/400 registry. Once the service principal is added to the Windows server (using ktpass), Windows expects **all machines** in that domain that are **capable** of using Kerberos to use Kerberos to authenticate to NetServer. If the Windows users are not mapped to a valid target OS/400 profile they will be denied access. To enable NetServer to allow SSO as well as your legacy connections you need to apply the following PTFs:

- 5722SS1-SI08801
- 5722999-MF30596
- 5722999-MF30597
- 5722999-MF31224

You will also need to know what the operating systems for the connecting clients are. Each of the supported clients, Windows 2000 and Windows XP, have different keytab entries that will need to be created manually. You may have selected to have the entries automatically created in Step 6 of 7.1.1, “Setting up Network Authentication Service with iSeries Navigator wizard” on page 99. If that is the case you may skip 8.4.5, “Creating the key tables on the iSeries server” on page 158.

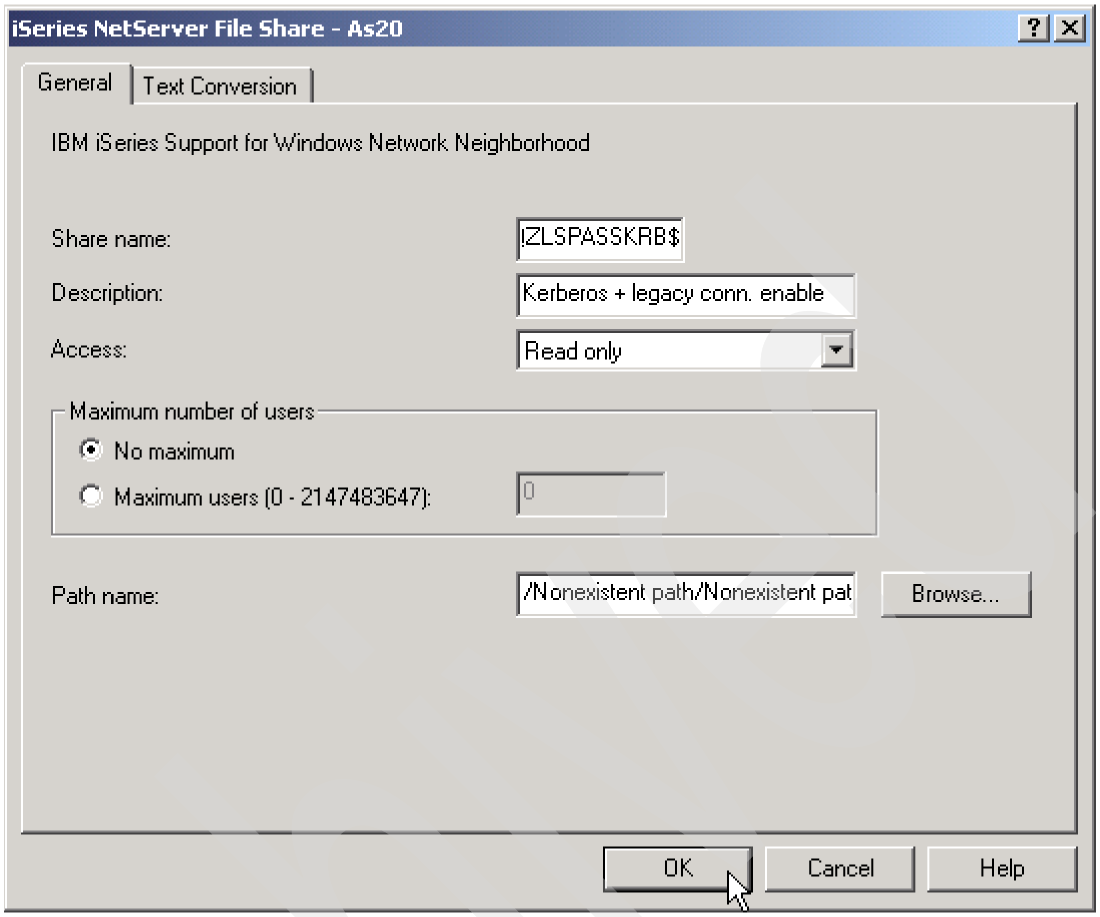
Restriction: iSeries NetServer only supports Version 5 (V5) of Kerberos. The client operating systems must also support that version or they will not be able to connect using Kerberos authentication.

8.4.2 Preparing NetServer for parallel use of SSO and legacy connection

Important: Do NOT proceed with this section until you read and understand the implications described in the previous section, 8.4.1, “Getting ready” on page 153.

In the first step of enabling Netserver to allow both types of signons you will need to create a dummy NetServer file share named QZLSPASSKRB$. To accomplish this follow these steps:

1. In the left panel of the iSeries Navigator expand your iSeries **Server -> Network -> Servers -> TCP/IP**.
2. In the right panel on the double-click **iSeries NetServer**.
3. iSeries NetServer window opens. Select **Shared Objects** in the left panel and right-click Shared Objects in the left panel; then select **New -> File** in the local menu.
4. In the iSeries File Share window (Figure 8-19) fill in the fields as follows:
   - Enter `QZLSPASSKRB$` in the Share name field.
   - Optionally enter a description.
   - Enter a nonexistent path in the Path name field (not to open any existing path for connection).
   - Click OK.


iSeries NetServer File Share - As20

General | Text Conversion

IBM iSeries Support for Windows Network Neighborhood

Share name: ZLSPASSKRB$

Description: Kerberos + legacy conn. enable

Access: Read only

Maximum number of users

(●) No maximum

( ) Maximum users (0 - 2147483647): 0

Path name: /Nonexistent path/Nonexistent pat

Figure 8-19 Create dummy file share QZLSPASSKRB$

With this configuration you will be able to make both single signon and your legacy connections. If you ever need to disable the feature, just remove the `QZLSPASSKRB$` file share.

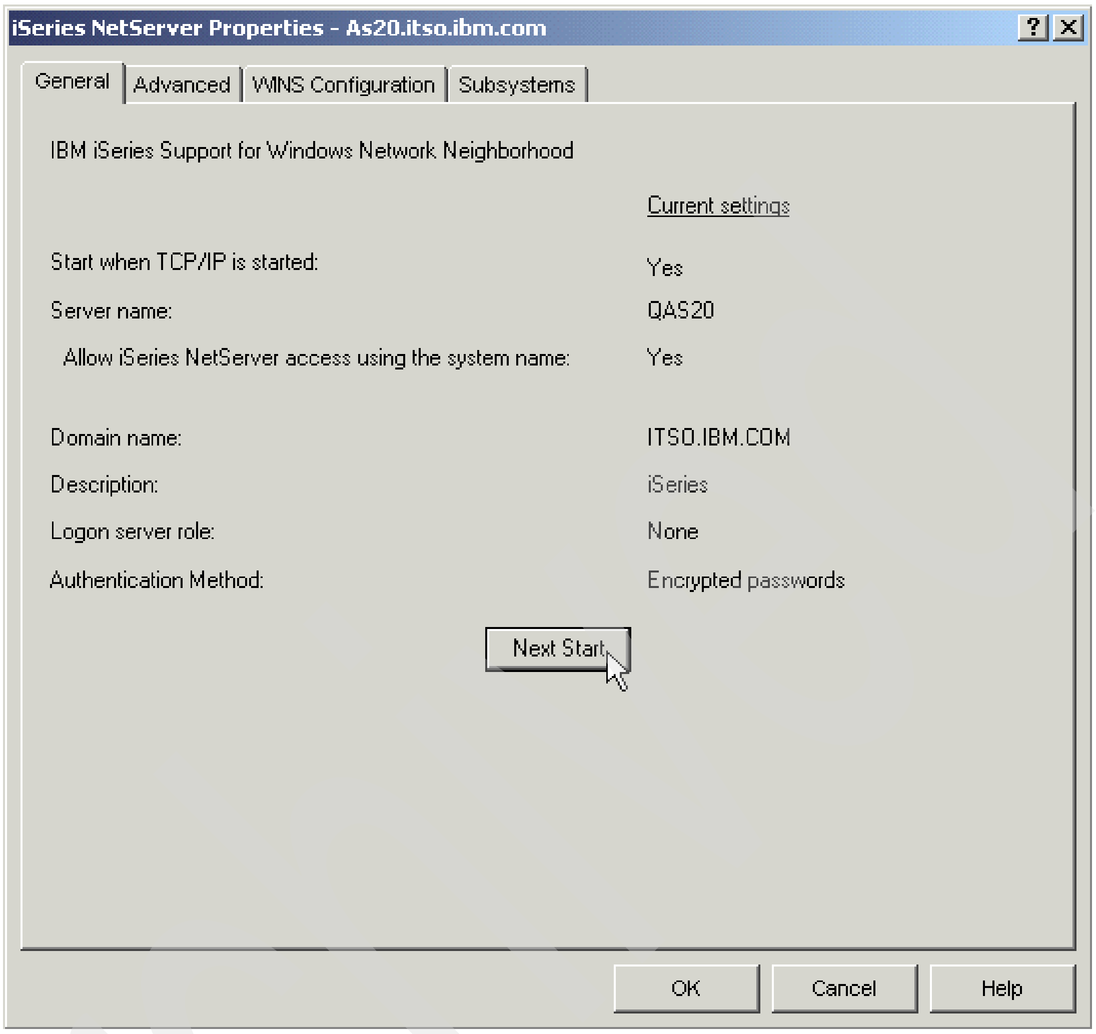
8.4.3 Checking and setting up NetServer properties

Now you need to check and set a few NetServer properties in the iSeries Navigator.

Important: Changes to NetServer require that the server be restarted. Notice in Figure 8-20, the Netserver properties window, by clicking the *Next Start* button you can make configuration changes that will not take effect until the server is restarted.

Perform the following steps:

1. In the left panel of the iSeries Navigator expand your iSeries Server -> **Network -> Servers** -> **TCP/IP**.
2. Right-click **iSeries NetServer** in the panel on the right; then select **Properties** in the local menu. The iSeries NetServer Properties window opens (Figure 8-20).



IBM iSeries Support for Windows Network Neighborhood

	Current settings
Start when TCP/IP is started:	Yes
Server name:	QAS20
Allow iSeries NetServer access using the system name:	Yes
Domain name:	ITSO.IBM.COM
Description:	iSeries
Logon server role:	None
Authentication Method:	Encrypted passwords

Figure 8-20 NetServer Properties

3. Click the **Next Start** button to reconfigure the server's parameters the next time the server is started.
4. The iSeries NetServer General Next Start window appears (Figure 8-21).

   Verify that the following values are correct:

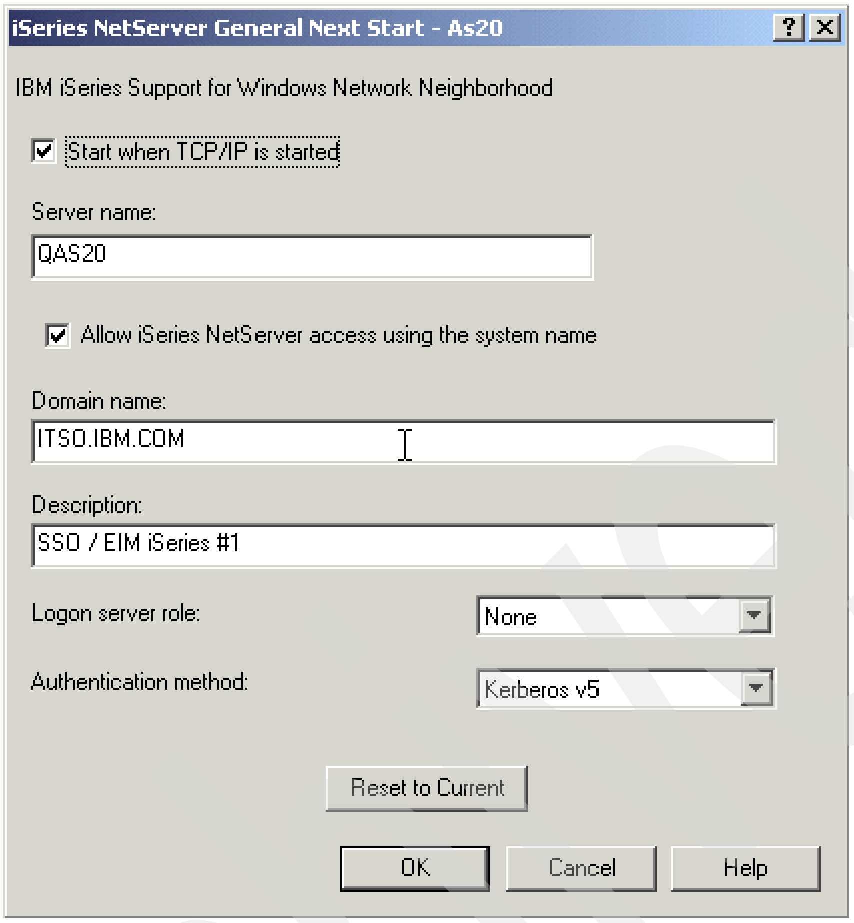
   – Server name (The default server name is the system name with the letter Q preceding it).

   – The check box labelled *Allow iSeries NetServer access using the system name* is optional. The commands to create a Kerberos principal for the system will be provided.

   – The Domain name from the NetServer properties should agree (except for the case) with the Domain name set in your TCP/IP configuration (option 12 of the OS/400 command Configure TCP/IP - CFGTCP). If they do not change the NetServer properties to match the TCP/IP configuration.

   – Select **Kerberos V5** in the Authentication method combo box.

   Click **OK**.


iSeries NetServer General Next Start - As20

IBM iSeries Support for Windows Network Neighborhood

[✓] Start when TCP/IP is started

Server name: QAS20

[✓] Allow iSeries NetServer access using the system name

Domain name: ITSO.IBM.COM

Description: SSO / EIM iSeries #1

Logon server role: None

Authentication method: Kerberos v5

Figure 8-21 NetServer properties - Next start

5. If you are ready to proceed with the configuration you may stop and restart NetServer now.
   – In right panel of iSeries Navigator, right-click **iSeries NetServer** and select **Stop**.
   – Verify that the server has stopped by refreshing the screen (F5).
   – Right-click **iSeries NetServer** again and select **Start**..

8.4.4 Creating the NetServer Kerberos principals

The next step is to create service principals for NetServer. Remember, not only users are principals, servers are also principals. With that in mind there are two issues that need to be addressed at this point.

First, how will your clients connect to the iSeries NetServer? We set up three different methods (principals) for the clients to connect, the fully qualified name, the IP address, and the NetServer name. Keep in mind that if the fully qualified name or the NetServer name are used the name must resolve to the iSeries.

The second point that needs attention is the clients on your network. Windows 2000 and Windows XP both support Kerberos V5 but they require different user names be defined for the principals to be created. Windows 2000 systems look for a service principal that the user name portion of the principal has been defined as *HOST*. Clients running the XP operating system seek the user name to be *cifs*. The following are examples of the different principals we created:

- A principal used to connect with the iSeries *fully qualified host name* (as20.itso.ibm.com in our example, you have to substitute your own value):

```
HOST/as20.itso.ibm.com@ITSO.IBM.COM
cifs/as20.itso.ibm.com@ITSO.IBM.COM
```

- A principal used to connect with the *IP address* (10.100.1.3 in our example, you have to substitute your own value):

```
HOST/10.100.1.3@ITSO.IBM.COM
cifs/10.100.1.3@ITSO.IBM.COM
```

- A principal used to connect with the *server name of the NetServer* (qas20 in our example, you have to substitute your own value):

```
HOST/qas20@ITSO.IBM.COM
cifs/qas20.ITSO.IBM.COM
```

Important: Both the *HOST* and *cifs* user names are case sensitive and must be entered in the Active Directory, ktpass command, and the iSeries key table as shown in the examples.

Creating new user accounts in Active Directory

We came up with a simplified way to identify each of the three methods we may use to connect with NetServer:

- `as20ns - for` the connection with the iSeries fully qualified host name
- `as20nsip -` for the connection with the IP address
- `as20nsq -` for the connection with the NetServer server name

We will create a user account in Active Directory following the same steps used in "Creating Active Directory account for your iSeries server" on page 105. When you create these users you will need to specify the exact password that will be used for the key table entry on the iSeries, and in the ktpass command.

Associate the Kerberos principal with the Active Directory account

Associate the Microsoft Active Directory accounts with the Kerberos principals using the following three commands at your Window 2000 server:

```
ktpass -princ HOST/as20.itso.ibm.com@ITSO.IBM.COM -mapuser as20nst -pass krbpwd
-mapOp set

ktpass -princ HOST/10.100.1.3@ITSO.IBM.COM -mapuser as20nsip -pass krbpwd -mapOp
set

ktpass -princ HOST/qas20@ITSO.IBM.COM -mapuser as20nsq -pass krbpwd -mapOp set
```

Tip: The `-mapOp set` is used to overwrite errors rather than deleting and re-entering all of your information (the `O` is an uppercase letter o).

8.4.5 Creating the key tables on the iSeries server

For each of the three principals you just created in Kerberos on Windows 2000 server you now need to create a key in the default key table of the Network Authentication Service on your iSeries server. To accomplish this, perform the following steps.

1. Enter the Qshell environment on your iSeries server, using the following OS/400 command:

   ```
   QSH
   ```

2. Create the three keys in the default Network Authentication Service key table using the following three Qshell commands:

   ```
   keytab add HOST/as20.itso.ibm.com@ITSO.IBM.COM -p krbpwd
   keytab add HOST/qas20@ITSO.IBM.COM            -p krbpwd
   keytab add HOST/10.100.1.3@ITSO.IBM.COM       -p krbpwd
   ```

   Again, the password specified in the -p parameter comes has to agree with the password you specified in the preceding steps for creating user accounts and associating the accounts with the Kerberos principal on the Windows 2000 Server.

3. List the Kerberos key table using the following Qshell command:

   ```
   keytab list
   ```

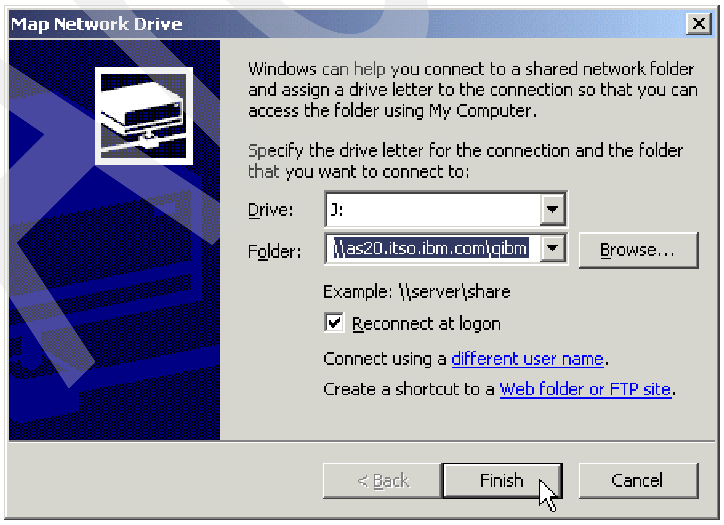
   This will produce a list of all the key table entries in the default key table. You will see one entry for each encryption level supported.

8.4.6 Verifying single signon with the NetServer

To verify that you have connected to the Netserver and your user has been mapped to an OS/400 profile follow these steps on your Windows 2000 client:

1. Sign on to the Windows domain as the test user, and to better demonstrate EIM, having different user ID for the iseries server, as you did to verify the signon with the iSeries Navigator and with the 5250 emulation.
2. Start Windows Explorer and in its menu select **Tools -> Map Network Drive**.
3. In the Map Network Drive window (Figure 8-22) fill in the folder name as follows:
   - **server**: specify fully qualified name of your iSeries server - \\as20.itso.ibm.com in our example.
   - **share**: specify name of a share, accessible for your test user as defined in the NetServer file shares - `\qibm` in our example.

   Then click **Finish**.


Map Network Drive

Windows can help you connect to a shared network folder and assign a drive letter to the connection so that you can access the folder using My Computer.

Specify the drive letter for the connection and the folder that you want to connect to:

Drive: J:

Folder: \\as20.itso.ibm.com\qibm

Example: \\server\share

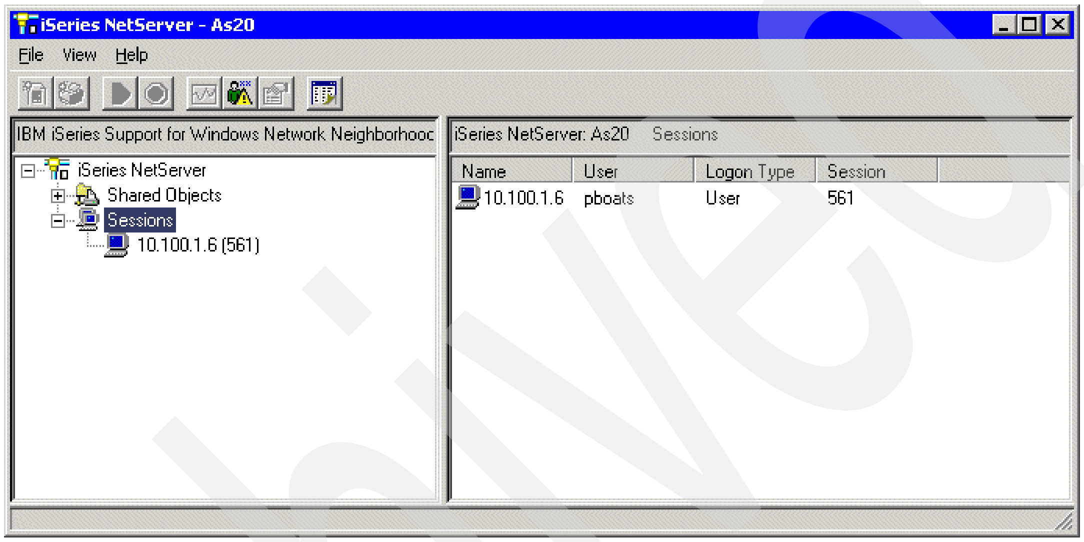
[✓] Reconnect at logon

Connect using a different user name.

Create a shortcut to a Web folder or FTP site.

Figure 8-22 Map Network Drive

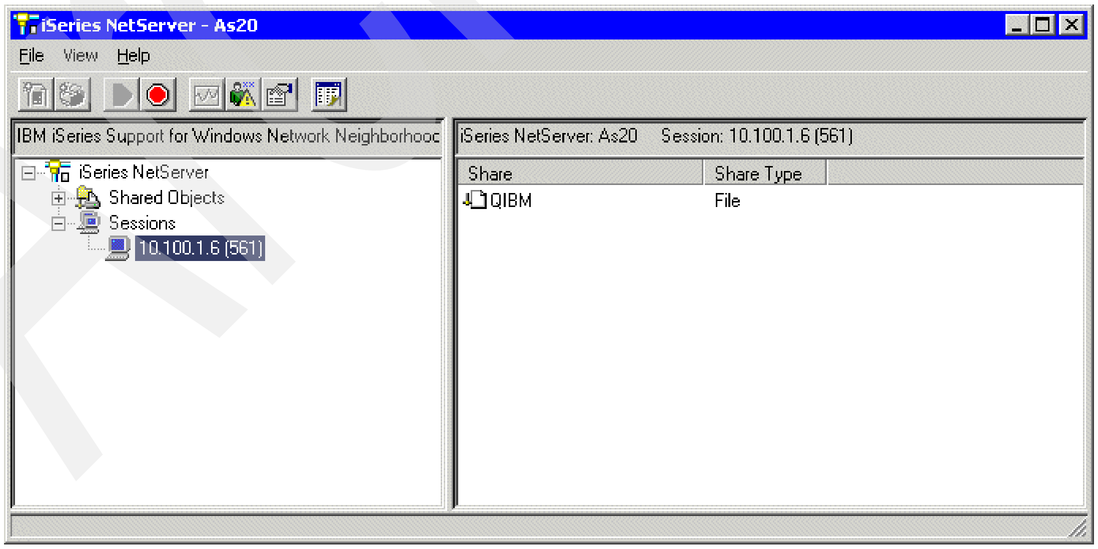
4. Note that you are not challenged for a signon and a window opens with the contents of the mapped drive.
5. To verify which user ID was used to connect:
   – Connect to the iSeries navigator with QSECOFR authorities.
   – Select **Network** -> **Servers -> TCP/IP**.
   – Double-click **iSeries NetServer**.
   – In the left panel of the iSeries NetServer window select **sessions**; you see the sessions in the panel to the right, as shown in Figure 8-23.



Name	User	Logon Type	Session
10.100.1.6	pboats	User	561

Figure 8-23 NetServer session

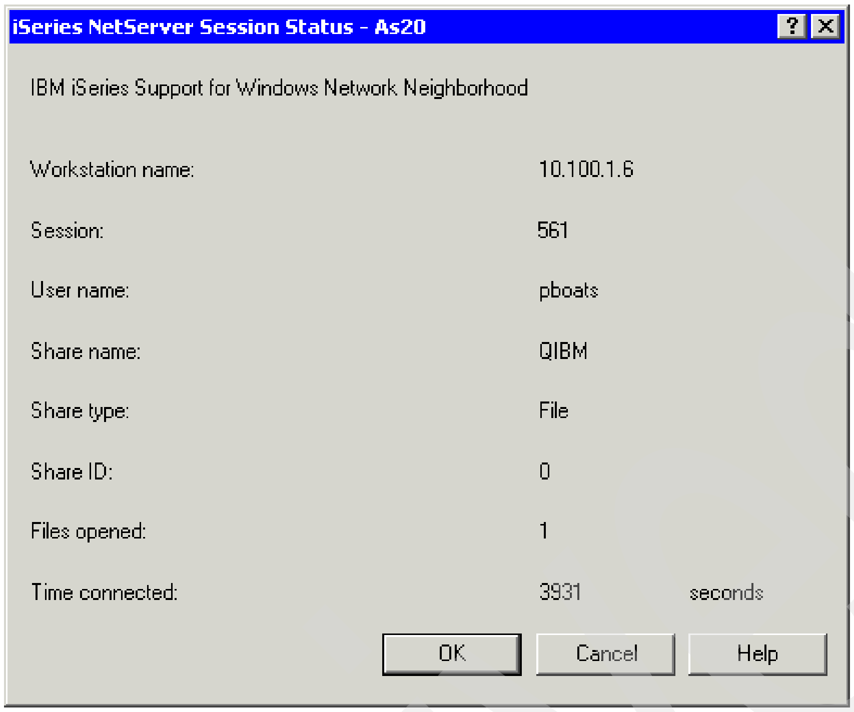
   – Now select the session in the left panel; you see the shares used in this session in the panel to the right, as shown in Figure 8-24.


iSeries NetServer - As20
File View Help
IBM iSeries Support for Windows Network Neighborhooc
iSeries NetServer
Shared Objects
Sessions
10.100.1.6 (561)
iSeries NetServer: As20 Session: 10.100.1.6 (561)

Share	Share Type
QIBM	File

Figure 8-24 Shares for a session

   – If you double-click the Share in the right panel you will get a detailed status window shown in Figure 8-25.

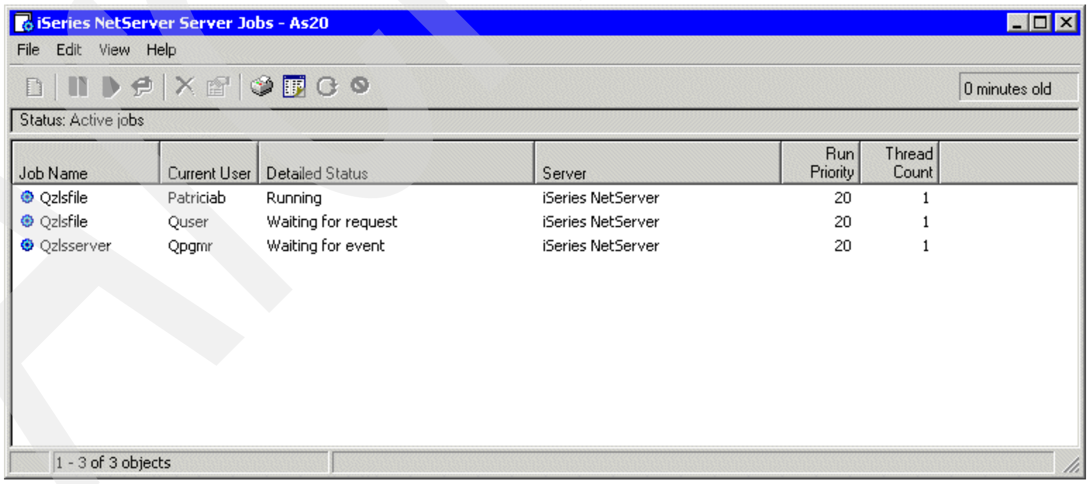


IBM iSeries Support for Windows Network Neighborhood

Workstation name:	10.100.1.6
Session:	561
User name:	pboats
Share name:	QIBM
Share type:	File
Share ID:	0
Files opened:	1
Time connected:	3931 seconds

Figure 8-25 Session status window

6. You may repeat the verification in steps 2 - 5 for the other two Kerberos principals.
7. Notice in Figure 8-25 that the user name is *pboats*, which is Patricia Boats' Windows user ID. To verify which user profile that job is running under on the iSeries go to the iSeries Navigator and expand your iSeries **Server -> Network -> Servers -> TCP/IP**. In the right panel, right-click **iSeries NetServer** and select **Server Jobs**. The window shown in Figure 8-26 will open. The first job listed is Patricia Boats' OS/400 profile PATRICIAB.



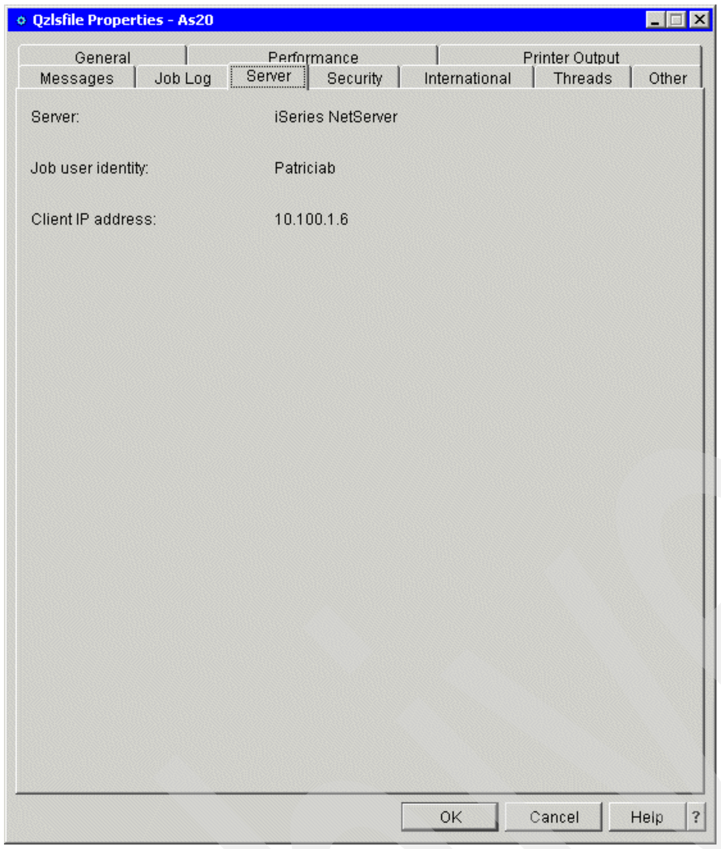
Status: Active jobs

Job Name	Current User	Detailed Status	Server	Run Priority	Thread Count
Qzlsfile	Patriciab	Running	iSeries NetServer	20	1
Qzlsfile	Quser	Waiting for request	iSeries NetServer	20	1
Qzlsserver	Qpgmr	Waiting for event	iSeries NetServer	20	1

1 - 3 of 3 objects

Figure 8-26 Mapped NetServer Job

8. To complete the verification you may double-click the job associated with that user. A properties window for the QZLSFILE job will open. Click the Server tab. In the Server window you can see that the client IP address is the same as the address listed in the session status window (Figure 8-25).



Server: iSeries NetServer

Job user identity: Patriciab

Client IP address: 10.100.1.6

Figure 8-27 Qzlsfile properties

This completes enabling the NetServer for single signon.

8.5 Enabling Domino Web Access for single signon and EIM

This section describes a single signon (SSO) solution for users of IBM @server iSeries Access for Web, running on a WebSphere Application Server, and Domino Web Access (formerly iNotes Web Access) connected with a Domino HTTP server. Users can access Lotus Domino functions (mail, calendar, to do list, etc.) with a browser through Domino Web Access without needing to enter their Domino user ID and Internet password. Your system administrator will control which users are allowed to use this function. A user must have a valid iSeries user ID and password to access iSeries Access for Web. This authentication will protect against unauthorized access to the Domino server and to OS/400.

This new solution provides two different SSO alternatives. In one method, the user does not have to have the same user ID for their iSeries user profile and for their Domino user ID. This implementation requires the system administrator to use IBM iSeries Enterprise Identity Mapping (EIM) to manage the mapping of the end users' OS/400 user profiles and their Domino user IDs. For the alternate implementation the users do have matching OS/400 user profiles and Domino user IDs, and there is less for the system administrator to manage. The administrator can decide to use either or can combine both implementations.

This section will:

- Give an overview of this SSO solution.
- List the prerequisite software needed to allow this solution to work.

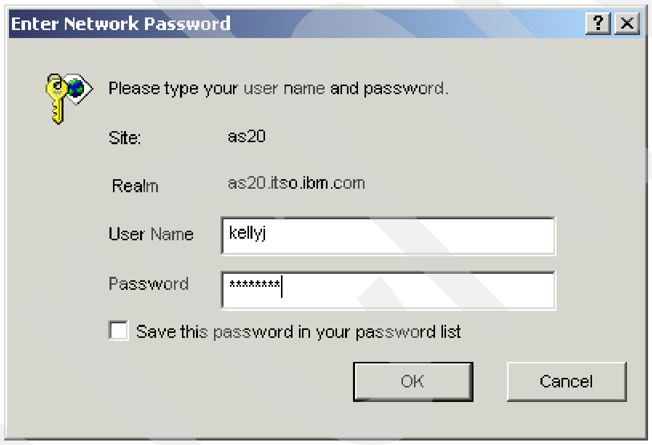
- Describe how to implement this SSO solution.
- Provide source code and a compiled service program to use or modify.
- Identify resources to learn more about each of the software products mentioned in this document.

Note: The SSO solution described in this document is not the only way to enable single sign on between WebSphere applications and Domino. There is a previously documented solution which uses the LTPA protocol, an LDAP server, and browser cookies to provide single sign on. The LTPA solution is a more general solution that can be applied to other applications. That solution is not described in this document.

The solution described in this document does not require the user to have the same user ID and password for both OS/400 and Domino, although they can if desired. It also does not require the user to manage the LDAP directory, as EIM does this for the user.

8.5.1 Overview

We'll start with a fictional end user, Kelly Johnson. Kelly uses her Internet Explorer Web browser to connect with her iSeries server. Using the URL for iSeries Access for Web, http://myiseries/webaccess/iWAHome. Figure 8-28 shows Kelly being prompted by the browser for her iSeries user ID and password. This information is passed via HTTP to the WebSphere server program, which authenticates the user ID and password to be valid.



Please type your user name and password.

Site: as20

Realm as20.itso.ibm.com

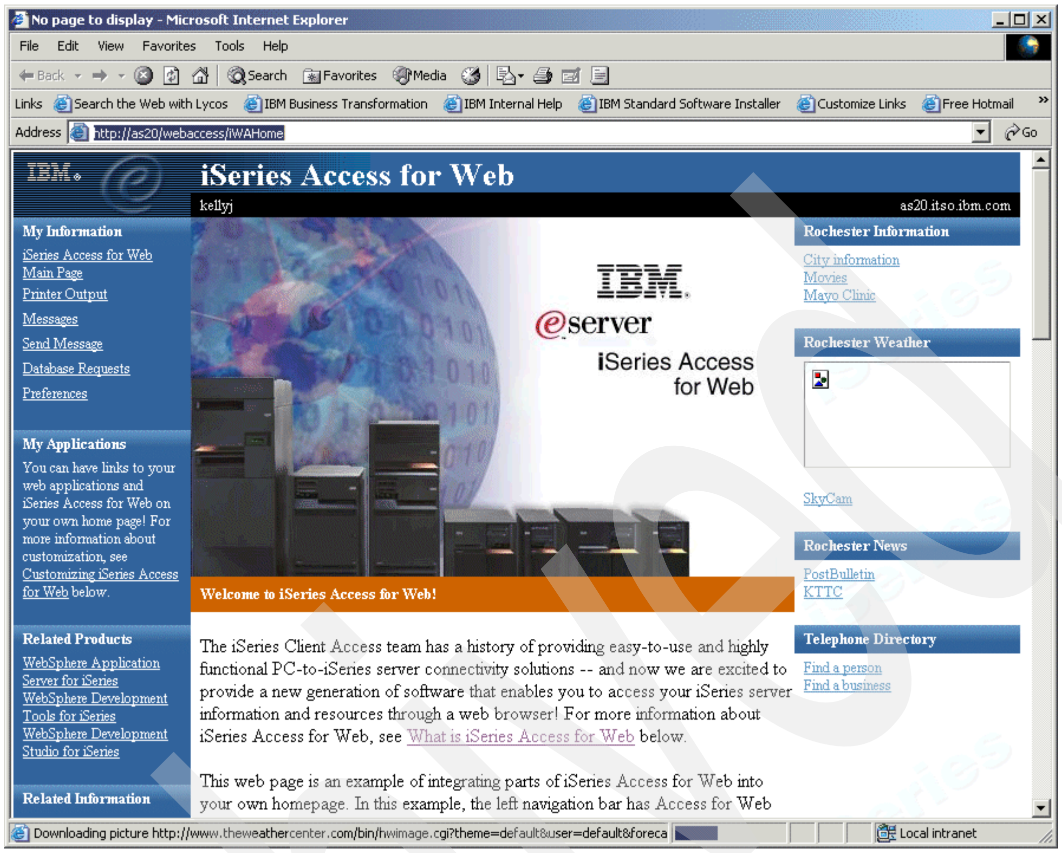
User Name kellyj

Password ********

Save this password in your password list

Figure 8-28 Prompt for iSeries user profile and password

Kelly now sees her iSeries Access for Web window on the browser (Figure 8-29).


No page to display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Links Search the Web with Lycos IBM Business Transformation IBM Internal Help IBM Standard Software Installer Customize Links Free Hotmail
Address http://as20/webaccess/iWAHome
Go

iSeries Access for Web

kellyj as20.itso.ibm.com

My Information
iSeries Access for Web Main Page
Printer Output
Messages
Send Message
Database Requests
Preferences

My Applications
You can have links to your web applications and iSeries Access for Web on your own home page! For more information about customization, see Customizing iSeries Access for Web below.

Related Products
WebSphere Application Server for iSeries
WebSphere Development Tools for iSeries
WebSphere Development Studio for iSeries

Related Information

IBM eserver iSeries Access for Web

Welcome to iSeries Access for Web!

The iSeries Client Access team has a history of providing easy-to-use and highly functional PC-to-iSeries server connectivity solutions -- and now we are excited to provide a new generation of software that enables you to access your iSeries server information and resources through a web browser! For more information about iSeries Access for Web, see What is iSeries Access for Web below.

This web page is an example of integrating parts of iSeries Access for Web into your own homepage. In this example, the left navigation bar has Access for Web

Rochester Information
City information
Movies
Mayo Clinic

Rochester Weather
SkyCam

Rochester News
PostBulletin
KTTC

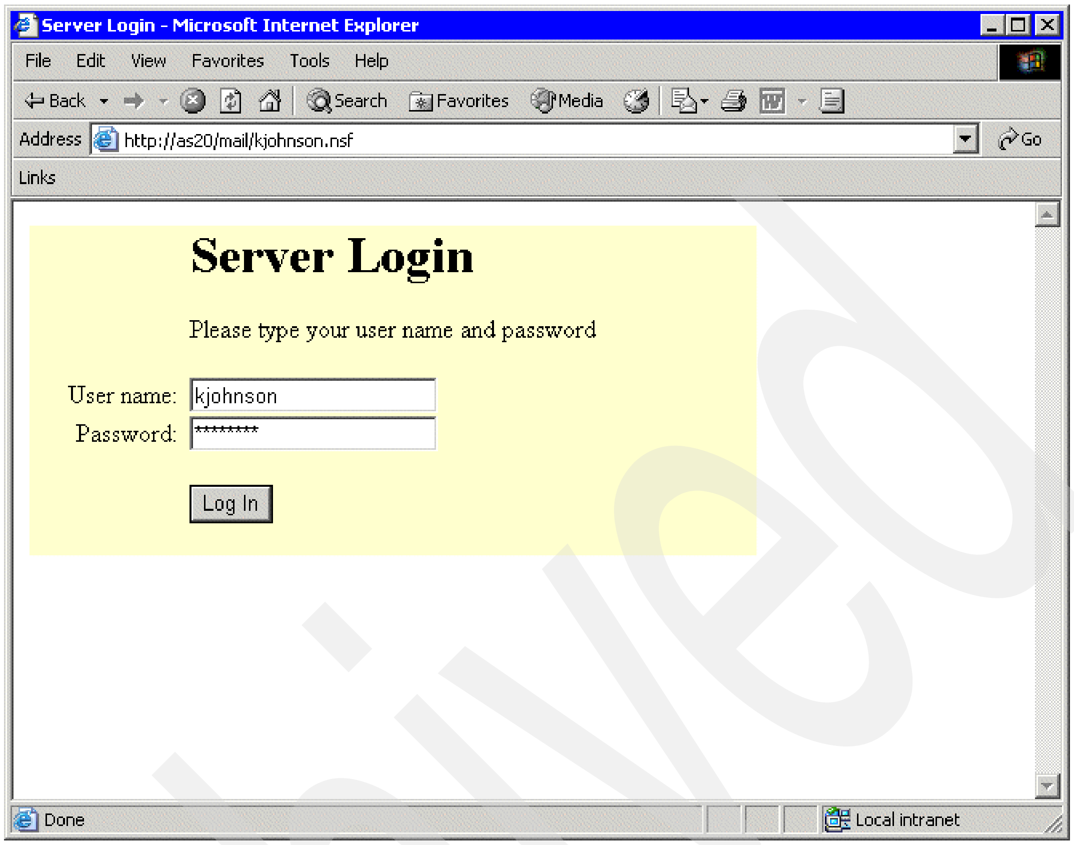
Telephone Directory
Find a person
Find a business

Downloading picture http://www.theweathercenter.com/bin/hwimage.cgi?theme=default&user=default&foreca

Local intranet

Figure 8-29 iSeries Access for Web welcome page

When Kelly is ready to see her e-mail, she directs the browser to the URL that brings up Domino Web Access: http://myiseries/mail/kjohnson.nsf. Currently, Kelly would be prompted for her Domino user ID and the Internet password associated with her Domino user ID (Figure 8-30).


Server Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://as20/mail/kjohnson.nsf
Go
Links
Done
Local intranet

Server Login

Please type your user name and password

User name: kjohnson

Password: *********

Log In

Figure 8-30 HTML prompt issued when SSO is not enabled

After entering these values, her Domino Web Access welcome window appears on the browser. However with this SSO solution enabled, Kelly does not get prompted for a Domino user ID and password – instead the Welcome window is presented (Figure 8-31) even though Kelly's iSeries user ID (KELLYJ) and Kelly's Domino user ID (kjohnson) are different.

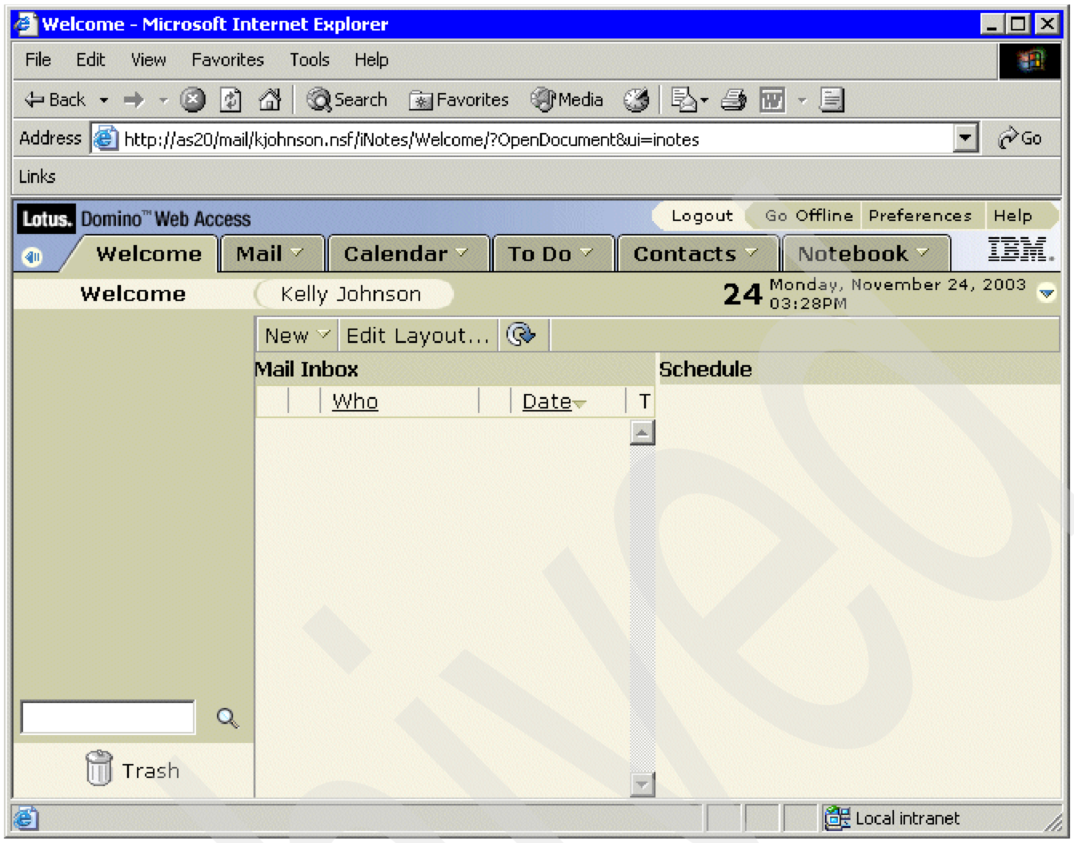

Welcome - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://as20/mail/kjohnson.nsf/iNotes/Welcome/?OpenDocument&ui=inotes
Go
Links
Lotus. Domino™ Web Access
Logout Go Offline Preferences Help
Welcome Mail Calendar To Do Contacts Notebook
IBM.
Welcome Kelly Johnson
24 Monday, November 24, 2003 03:28PM
New Edit Layout...
Mail Inbox
Schedule
Who Date T
Trash
Local intranet

Figure 8-31 Domino Web Access Welcome page

Communication between the browser and the server takes place using HTTP. For this solution, the HTTP server that runs in Domino is used. This HTTP server has a documented exit that a user program can monitor the HTTP data and respond to it when appropriate. This exit program is called a DSAPI (Domino Web Server Application Program Interface) exit. DSAPI (pronounced dis-sappy) exit programs must be registered in the Domino server's server document. More than one DSAPI exit program may be registered. For this SSO solution, the iSeries Access for Web product provides a DSAPI exit so that it can run using the Domino HTTP server, and the system administrator provides another DSAPI exit program which handles the SSO processing. We have provided a working SSO DSAPI exit program (written in C) that you can use as is, or modify to fit your needs. The Domino HTTP server and the WebSphere instance with Access for Web must be using the same HTTP address and port, and must also be using the same Web realm. (You can specify your Web realm in your Domino server document.) These are determined when the servers are configured. This gets both servers talking on the same path, so their DSAPI exit programs will see the same data.

This solution also uses Enterprise Identity Mapping (EIM) which provides the capability to have SSO work with different user IDs. We'll call this plan A. This solution can also be implemented without using EIM – in which case the users must have the same user ID for iSeries and for Domino. We'll call this plan B.

EIM uses an LDAP directory to store its information. This can be any LDAP directory on any IBM @server. EIM completely manages this LDAP directory, so that the system administrator does not need to. The administrator need only provide the Domino server with the user ID and password of the LDAP administrator or another LDAP ID that is a member of the LDAP Administration group (Any further reference to the LDAP administrator in this section refers to any member of the LDAP administration group). The LDAP administrator ID is stored in the

Domino server's notes.ini file. The LDAP administrator password is encrypted in a Validation List Object (VLDL). The name and library of the VLDL are also stored in the notes.ini file. The code provided with this solution uses an alias of "400users" for the source association with each EIM identifier, and an alias of "dominousers" for the target association with each identifier. (In simple terms, this code looks up the OS/400 user ID, and returns the corresponding Domino user ID.) The system administrator can choose other alias names if they want.

iSeries administrator tasks

- The administrator configures an instance of WebSphere to use the Domino Server's HTTP task. They will be sharing the same Internet address and port.
- The administrator has their Domino server set up to use the same Web realm used by iSeries Access for Web.
- The administrator also puts the names of the DSAPI exit programs into the Domino Server's Server document.
- Because EIM uses LDAP, the administrator puts the LDAP administrator user ID in the notes.ini file. (Remember this LDAP ID can be any member of the LDAP administration group.) The administrator also creates a VLDL containing the LDAP administrator password in encrypted form. The name and library of this VLDL object are also put into the notes.ini file. A working sample command and program to put the password into the VLDL are included with this solution.
- Using EIM, the administrator defines the iSeries and Domino user IDs for each user who will be using the SSO support. The user IDs do not have to match.
- Passwords are NOT included in the EIM data. This solution relies on the OS/400 user ID and password being valid. If the user is valid for OS/400, then EIM provides the Domino User ID and assumes it is also valid. An OS/400 user who is not a valid Domino user would not have an identity in the EIM list of identifiers.

End user tasks

- The end user, Kelly, signs on to iSeries Access for Web as usual. She uses her OS/400 user ID and password. There is no change to this product or its use.
- Kelly then enters the URL to open Domino Web Access on her Web browser. That's it. Kelly's Domino Web Access welcome window is displayed in her Web browser.

What happens in the system when Kelly enters the Domino Web Access URL

- The SSO DSAPI exit program sees the attempted access to the Domino server, and takes the user ID and password provided by the browser (Kelly's OS/400 user ID and password which is in the HTTP headers that the exit program can look at when using basic authentication). This information is in the headers because the same IP address and port are being used, and because the same realm is being used.
- The exit program authenticates that Kelly has a valid OS/400 user profile.
- The exit program accesses EIM with the iSeries (source) user ID.
- EIM returns Kelly's corresponding Domino (target) user ID. (EIM uses the LDAP administrator user ID and password to look up the correct Domino user ID).
- The Domino user ID is passed back up through HTTP to the browser with an indication that the user has been authenticated. Since the browser does not receive a 401 it does not prompt for a Domino user ID and Internet password. It will present the Domino Web Access welcome window.

Note: No Domino Web Access Internet password was used. Since Kelly had a valid OS/400 ID that was checked again when she entered Domino Web Access, she does not get authenticated again. If Kelly did not have a corresponding Domino user ID stored in EIM, she would be prompted for her Domino user ID and Internet password – just as when SSO had not been set up.

- Plan B: If the administrator did not set up the VLDL and EIM, and wants to use this solution where user IDs must match, then once the iSeries user ID and password have been authenticated by the exit program, the exit program returns a Domino user ID that is the same as the iSeries user ID. If this is a valid Domino user ID, the user will see the Domino Web Access welcome window without being prompted for a Domino user ID and Internet password. If this is not a valid Domino user ID (if Domino can not authenticate this user), the user will see the prompt requiring them to enter a valid Domino user ID and Internet password.

8.5.2 Prerequisites

- The Domino server must be at Version 6.0.0, at a minimum (our scenario is 6.5).
- The Domino HTTP server must be used.
- OS/400 must be at V5R2 or later.
- Series Access for Windows V5R2 (5722-XE1).
- iSeries Access for Web V5R2 (5722-XH2).
- The system value QRETSVRSEC must be set to 1. (Not needed for plan B.)
- iSeries Access for Web must be set up on each PC that will be used.
- One of the following WebSphere products:
  - WebSphere Application Server V5 - Express for iSeries
  - WebSphere Application Server V5 for iSeries ("Base Edition")
  - WebSphere Application Server V4 Advanced Edition (AE) for iSeries, with latest PTFs
  - WebSphere Application Server V4 Advanced Edition Single Server (AEs) for iSeries

Restriction: The preceding list of WebSphere products are the only servers that support both iSeries Access for Web and the Domino HTTP server connection.

8.5.3 Set up

To setup Network Authentication Service and Enterprise Identity Mapping, refer to Chapter 7, “Enabling Network Authentication Service and Enterprise Identity Mapping” on page 97.

This solution requires HTTP to use basic authentication. It will run using either an unencrypted HTTP connection, or using SSL to encrypt transmitted data.

Re-compiling of the C module and service program are only needed if you change the source from what has been provided. Instructions for this are at the end of this document.

Acquiring iSeries Access for Web

V5R2 iSeries Access for Web (licensed program product 5722-XH2) is delivered with the iSeries Access Family (5722-XW1) of products. The iSeries Access Family and iSeries Access for Web products are included on the Keyed Stamped Media, which provides customers a 70-day evaluation period to use the iSeries Access for Web functions. Using iSeries Access for Web after the 70-day evaluation period requires that a user has purchased licenses for the iSeries Access Family. To find out how to order or upgrade iSeries Access for Web, refer to the iSeries Access Ordering Web page at:

http://www.ibm.com/eserver/iseries/access/caorder.htm

For information on installing and configuring iSeries Access for Web with these Web application servers refer to the iSeries Access for Web home page at:

http://www.ibm.com/eserver/iseries/access/web

Domino Server Setup

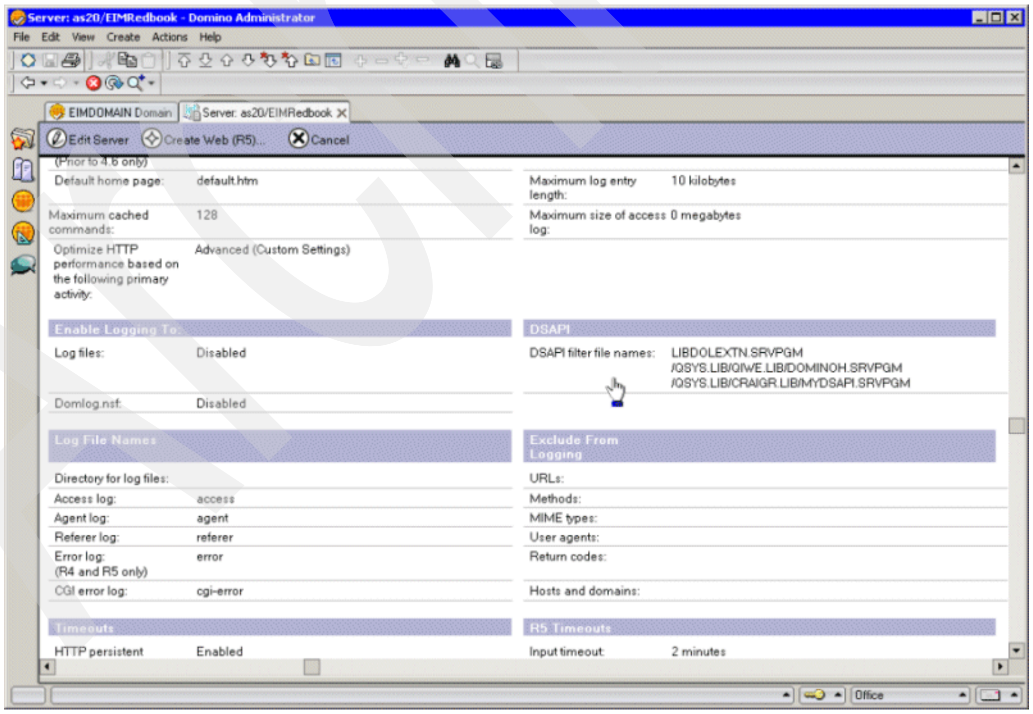
To make the necessary changes, you need to open the Domino Administrator and navigate to your server. Click the **Configuration** tab and in the navigation pane (left pane) select **All Server Documents**. Expand the domain that you want to work with and double-click the server name. In the next window select the **Internet Protocols** tab.

Note: Changes to the notes.ini file and changes to the server document both require that the Domino server be ended and restarted before those changes will take effect.

Server document changes

- In the **Internet Protocols** window select the **HTTP** tab. Locate the DSAPI section, Figure 8-32, and enter the names of the DSAPI exit programs:
  - /QSYS.LIB/QIWE.LIB/DOMINOH.SRVPGM
  - /QSYS.LIB/YourLibName.LIB/MYDSAPI.SRVPGM

  The first is for iSeries Access for Web to use this HTTP server (this is the DSAPI exit for WebSphere Express), and the second is the new SSO exit program.



(Prior to 4.6 only)			
Default home page:	default.htm	Maximum log entry length:	10 kilobytes
Maximum cached commands:	128	Maximum size of access log:	0 megabytes
Optimize HTTP performance based on the following primary activity:	Advanced (Custom Settings)		
Log files:	Disabled	DSAPI filter file names:	LIBDOLEXTN.SRVPGM /QSYS.LIB/QIWE.LIB/DOMINOH.SRVPGM /QSYS.LIB/CRAIGR.LIB/MYDSAPI.SRVPGM
Domlog.nsf:	Disabled		
Directory for log files:		URLs:	
Access log:	access	Methods:	
Agent log:	agent	MIME types:	
Referer log:	referer	User agents:	
Error log: (R4 and R5 only)	error	Return codes:	
CGI error log:	cgi-error	Hosts and domains:	
HTTP persistent	Enabled	Input timeout:	2 minutes

Figure 8-32 Adding the DSAPI exit programs

Note: The service program name above is MYDSAPI in the server document. The service program name is LIBMYDSAPI. The LIB part of the name is necessary. You need it on the service program name, but not in the DSAPI filter name entered in the server document.

- When you first connect your browser to iSeries Access for Web and get the prompt for the user ID and password, the name of the realm is listed, shown in Figure 8-33. You need to remember this realm name.

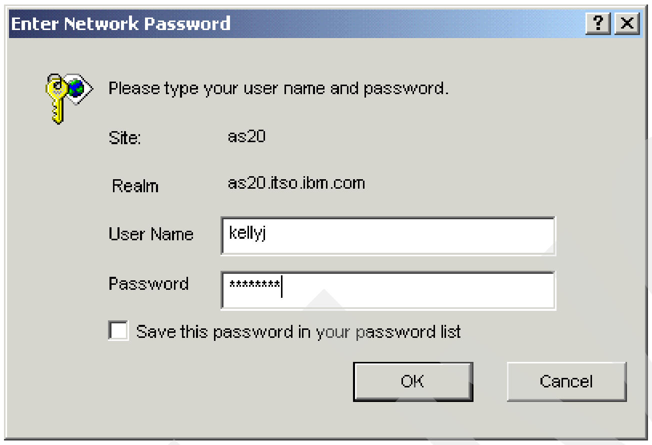
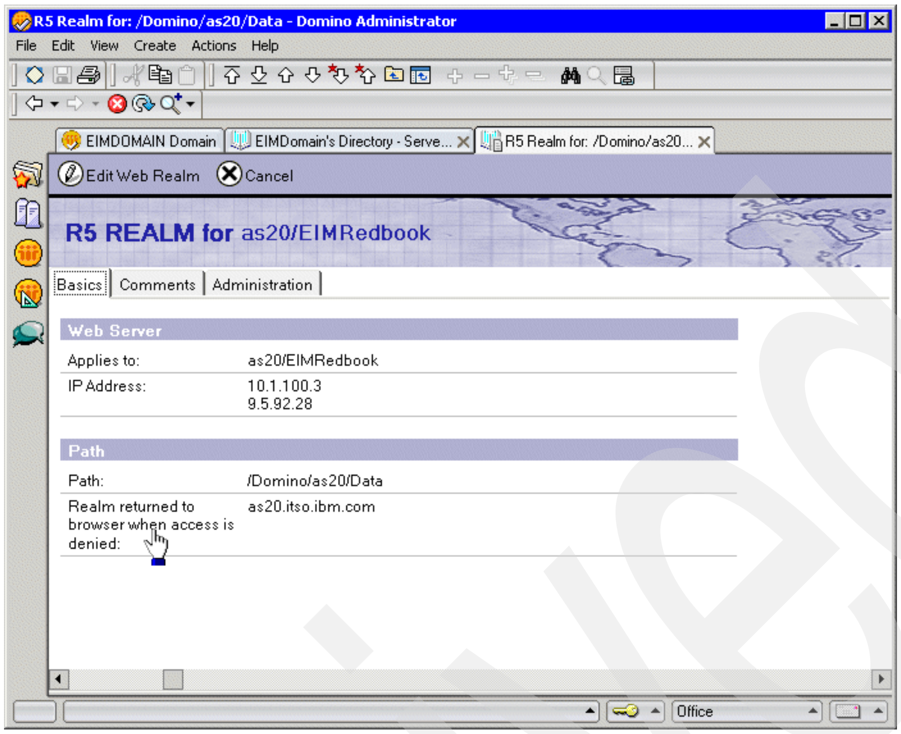


Figure 8-33 Realm listed on the logon prompt

- Click the **Domain** tab, in the left pane expand **Web**, and then click **Webserver Configurations**. You should see your server listed in the right pane, expand the server and verify whether there is currently a realm configured. If there is a realm listed double-click the realm name to open the description, it should look similar to Figure 8-34. If there is no realm configured double-click the Domino Server listed in the right pane. That will take you back to the **Basics** tab in the **All Server Documents** window. Above the server name, click the drop down menu labeled Create Web (R5) and select **Realm**.


R5 Realm for: /Domino/as20/Data - Domino Administrator
File Edit View Create Actions Help
EIMDOMAIN Domain
EIMDomain's Directory - Serve...
R5 Realm for: /Domino/as20...
Edit Web Realm
Cancel
Basics
Comments
Administration
Office

R5 REALM for as20/EIMRedbook

Web Server

Applies to:	as20/EIMRedbook
IP Address:	10.1.100.3 9.5.92.28

Path

Path:	/Domino/as20/Data
Realm returned to browser when access is denied:	as20.itso.ibm.com

Figure 8-34 Realm configuration

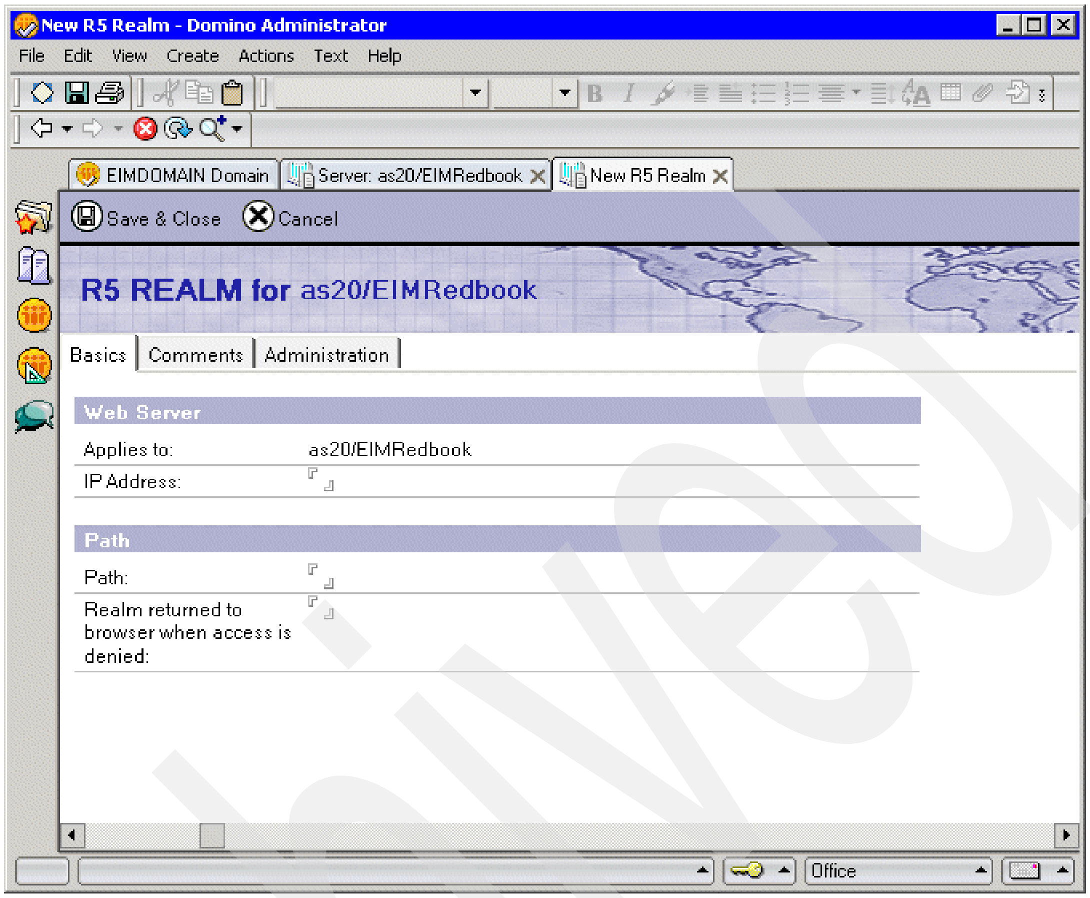
The New R5 Realm window shown Figure 8-35 in will open. Fill in the parameters as follows:

– In the **IP Address** parameter list the IP address that is used for the server to communicate with Notes clients. This is not necessarily the same IP address that the Domino Web Access browsers use to communicate with the server.

– The **Path** parameter will specify the Domino server's data directory.

– **The Realm returned to browser when access is denied** parameter is the string that is sent to the browser to specify the realm that the user is signing into when the HTTP server returns a 401 message.

– Save these changes to the server document.



R5 REALM for as20/EIMRedbook

Basics | Comments | Administration

Web Server

Applies to:	as20/EIMRedbook
IP Address:	

Path

Path:	
Realm returned to browser when access is denied:	

Figure 8-35 Window to configure a new realm

Notes.ini changes

Plan A: The following entries need to be added to the Domino server's notes.ini file. On an iSeries command line use the Work with Domino Servers (**wrkdomsvr**) command and use an option 13 to edit the notes.ini file.

These entries are case sensitive and must be entered exactly, the parameter after the = sign should be replaced with variables specific to your environment:

- Eimldapid=ldapadmin (for example, Eimldapid=cn=ROOT)
- Eimldapvldl=nameofvldl (for example, Eimldapvldl=MYVLDL)
- Eimldapvldllib=nameofvldllibrary (for example, Eimldapvldllib=MYLIB)

This Validation List object, or VLDL will contain the encrypted password for the LDAP administrator (Remember that this can be a member of the LDAP administration group). To keep this password secure, it is not kept in the clear here in the notes.ini file.

For each end user that will be using Domino Web Access, you do not have to create an Internet Password. Because authentication is done against the iSeries user ID and password, the Internet password is not needed.

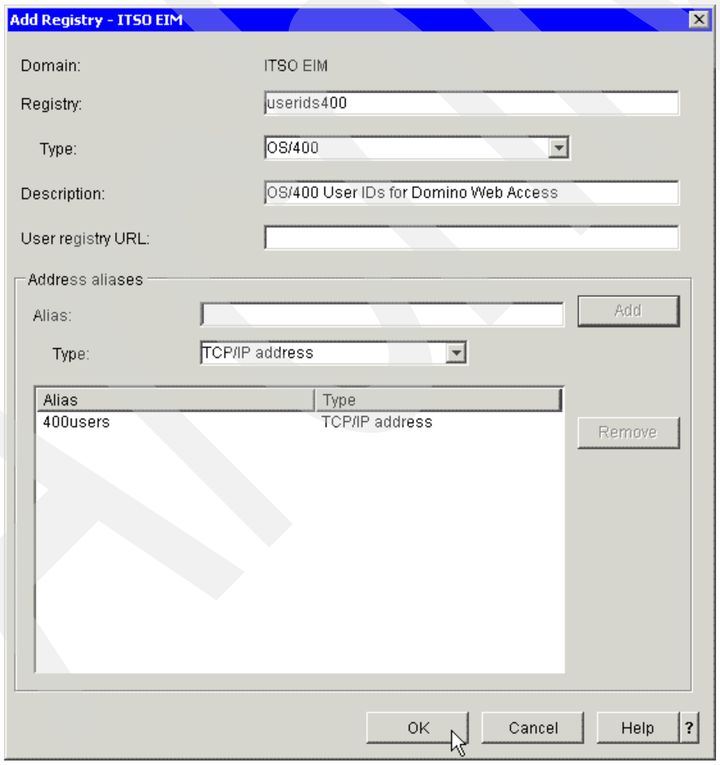
EIM Setup (Plan A only)

This scenario assumes that you have completed all of the steps in Chapter 7, "Enabling Network Authentication Service and Enterprise Identity Mapping" on page 97.

Connect to the iSeries using iSeries Navigator. Expand the **server name -> Network -> Enterprise Identity Mapping -> Domain Management**, then expand the domain you want to work with, and click **User Registries**.

To make your configuration look the same as ours you will have to create two new registries, one named userids400 (with and alias of 400users), and another named useridsdomino (with an alias of dominousers). These registries could be named anything; the important thing to note is the alias names. They must be the same as ours in order to use the program that was written for this scenario. That program references the 400users and dominousers alias names rather than the actual name of the registry. You may change the program to access your existing registries, or create aliases for your existing registries, if you choose not to create the new ones. To add the new registries follow these steps:

1. Right-click **User Registries** and select **Add Registry...**
2. In the Registry parameter use the name *userids400*.
3. In the Type parameter select OS/400.
4. You may include a description that will identify this registry as your Domino Web Access OS/400 profiles.
5. To make the scenario work as is you need to create an alias.
   a. In the Alias parameter type the name *400users*.
   b. Select the alias type TCP/IP addresses and click Add.
6. When your window looks like Figure 8-36 click OK.



Domain: ITSO EIM
Registry: userids400
Type: OS/400
Description: OS/400 User IDs for Domino Web Access
User registry URL:
Alias:
Type: TCP/IP address

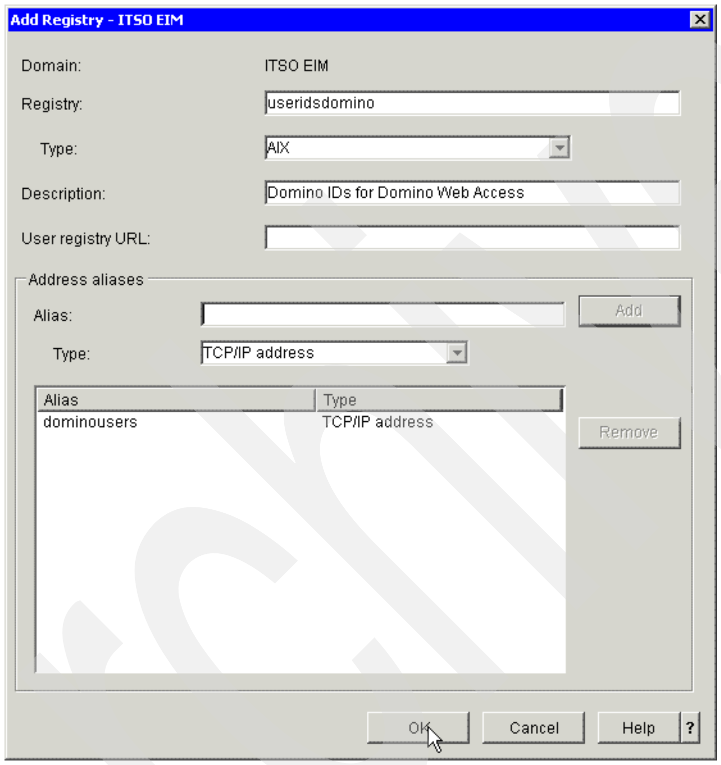
Alias	Type
400users	TCP/IP address

Figure 8-36 Adding the OS/400 user registry

Next we create the registry for Domino user IDs. This process will be similar to creating the registry for OS/400 users.

1. Right-click **User Registries** and select **Add Registry...**

2. In the Registry parameter use the name *userids domino*.
3. If your version of EIM does not allow selection of Domino as a type of user registry, you can use a different type which allows mixed case, such as AIX.
4. You may include a description that will identify this registry as your OS/400 profiles for Domino Web Access.
5. Again, to make the scenario work with the program provided you will need to create an alias.
   a. In the Alias parameter type the name *dominousers*.
   b. Select the alias type TCP/IP addresses and click Add.
6. When your window looks like Figure 8-37 click OK.



Domain: ITSO EIM
Registry: userids domino
Type: AIX
Description: Domino IDs for Domino Web Access
User registry URL:

Address aliases
Alias:
Type: TCP/IP address

Alias	Type
dominousers	TCP/IP address

Figure 8-37 Adding a Domino user ID registry

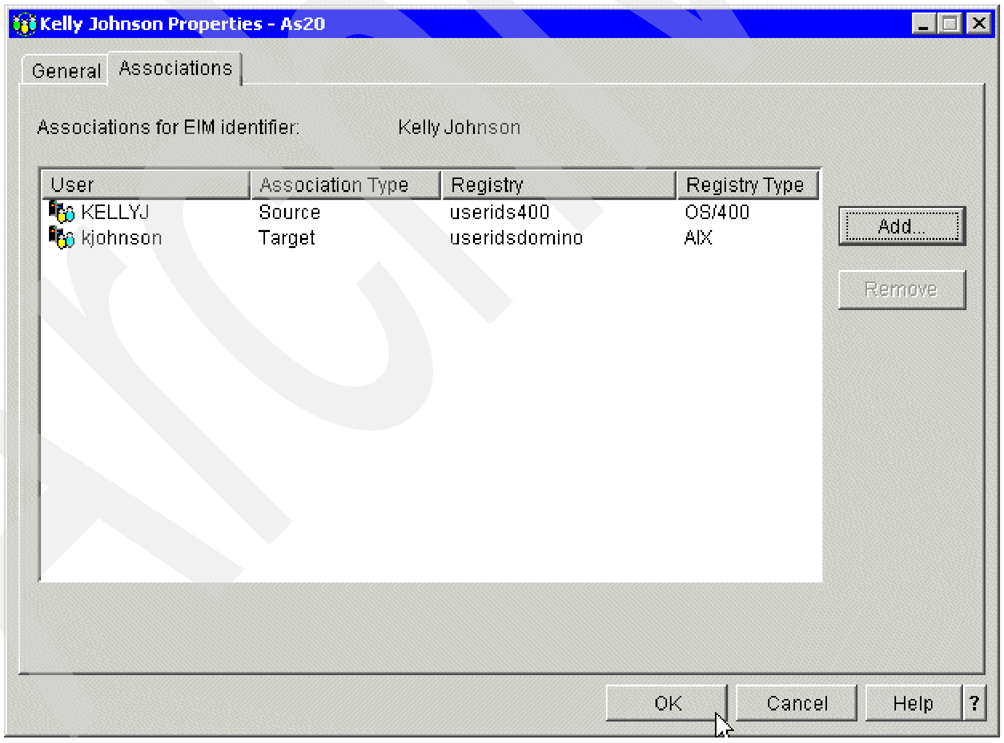
If your users do not have an identifier already you will have to create one each user that will be taking advantage of this SSO solution. Associations between the EIM identifier and OS/400 user profiles (userids400) should be source associations and those between the EIM identifier and the Domino user IDs (userids domino) should be target associations. Note that if you were previously using EIM for a different single sign-on application with Windows, there may already be a target association for the OS/400 user profile. An additional source association must be created for single signon in Domino. To create the EIM indentifier for Kelly Johnson, we followed these steps:

1. In iSeries Navigator expand your iSeries **Server name -> Network -> Enterprise Identity Mapping -> Domain Management**, then expand the domain you want to work with, and click **Identifiers**. Verify that Kelly Johnson does not already exist in the list of existing identifiers. If that identifier does not exist go to Step 2, if the identifier is already in the list proceed to Step 5.
2. Right-click **Identifiers** and select **New Identifier...**

3. In the Identifier parameter place the name of one of your users, we supplied the name Kelly Johnson.
4. If you want to fill in aliases at this point you may but it is not required for this scenario. When the parameters are filled in to your liking, click OK.

   You will see that the user is added in the right pane.
5. Right-click the user's Identifier and select **Properties**.
6. Click the **Associations** tab.
7. In the Associations window, click **Add**.
8. In the Add Associations box, next to the Registry field, select **Browse**.
9. When the Browse EIM Registries window opens, select **userids400** and click OK.
10. In the User parameter supply the user ID for the iSeries, in our case it is KELLYJ.
11. Finally select **Source** as the Association Type and click OK.
12. Next click the **Add** button again.
13. Select **Browse** again but this time choose the **useridsdomino** registry.
14. Fill in the users Domino user profile for the User parameter.
15. Select **Target** as the Association Type and then click OK.

When you click the Association tab in your new identifier's properties window the associations should look like Figure 8-38.



Associations for EIM identifier: Kelly Johnson

User	Association Type	Registry	Registry Type
KELLYJ	Source	userids400	OS/400
kjohnson	Target	useridsdomino	AIX

Figure 8-38 Associations to enable SSO to Domino Web Access

Validation List Object (Plan A only)

Create your Validation List Object using the CRTVLDL CL command. (You can use the DLTVLDL command to delete the Validation List Object.) For example: CRTVLDL VLDL(MYLIB/MYVLDL).

Putting the LDAP administrator's password into the Validation List Object takes a little more work. There is a sample C program that uses the Validation List Object API QsyAddValidationListEntry to fill in the Validation List Object. In the object, you need to put the password for the LDAP administrator, and also an Entry ID value. This is a text string that the DSAPI exit program will also use, to verify that the correct entry is being retrieved from the Validation List Object. In the sample program, we enter the Entry ID string 'My User ID' and an Entry ID length of 10. we also enter the data to be encrypted (this is the LDAP administrator password) 'phred', and the length of that data – 5 if the password is 'phred'.

Once the password is encrypted in the Validation List Object, only an application that is authorized to the Validation List Object, and knows the Entry ID will be able to extract the unencrypted password.

Change the Validation List Object owner so that the QNOTES user profile has *ALL authority to the object, and remove any other users.   This is done to keep the password value from being changed by others and to allow the SSO DSAPI exit program to be able to use the Validation List Object.

Along with the sample programs provided with this solution, there is a CL command, add to validation list (ADDTOVLDL), which takes the name and library of a Validation List Object as input. It also takes the LDAP administrator's password as input.   This command invokes a program, ADDCODE, which puts that password, encrypted, into the Validation List Object.

The system value QRETSVRSEC must be set to 1 (one) in order for the Validation List Object to be able to return the password unencrypted. (Plan A)

8.5.4 Downloading the source code

Prototype program source code can be downloaded from:

ftp://www.redbooks.ibm.com/redbooks/SG246975

The Domino Web Access for single signon and EIM source file are in the zip file labeled DominoWebAccess.zip. The zip file contains a save file named CRSAVF.SAVF which is a saved library that holds all the files needed to facilitate enabling Domino Web Access for single signon. You need to get the SAVF to your iSeries and restore it. If you are unsure of the process you can follow these steps:

1. To start the download go to the FTP site listed above and double-click the file named DominoWebAccess.zip.
2. Select the option to save the file to your PC rather than opening it. Since we will have to FTP the file later it is easiest just to save the file directly to your C: drive.
3. Once the download has completed we need to extract the SAVF from the zip file. Start the utility that you use to open zip files and extract the SAVF named CRSAVF to the C: drive.
4. On the iSeries command line you will need to pre-create a SAVF with that same name (CRSAVF) as the file on the PC using the follow syntax.

   ```
   CRTSAVF FILE(MYLIB/CRSAVF) TEXT('SAVF for Domino Web Access')
   ```

   (MYLIB can be any library that you want to store the file in.)
5. Now FTP the SAVF to your iSeries. Example 8-21 demonstrates the steps required to successfully FTP a SAVF to the iSeries.

Example 8-21   FTP syntax

```
C:\>ftp your.system.name
Connected to your.system.name.
220-QTCP at your.system.name.
```

```
220 Connection will close if idle more than 5 minutes.
User (your.system.name:(none)): glakner
331 Enter password.
Password:
230 GLAKNER logged on.
ftp> quote site na 0
250  Now using naming format "0".
ftp> cd MYLIB
250 "MYLIB" is current library.
ftp> bin
200 Representation type is binary IMAGE.
ftp> put crsavf.savf
200 PORT subcommand request successful.
150 Sending file to member SAVF in file CRSAVF in library MYLIB.
250 File transfer completed successfully.
ftp: 992640 bytes sent in 0.27Seconds 3676.44Kbytes/sec.
ftp> quit
221 QUIT subcommand received.
```

6. After the FTP completes display the savf to verify the contents using the command:

   DSPSAVF FILE(MYLIB/CRSAVF)

   The output should look like Example 8-22 and the second page should look like Example 8-23.

Example 8-22   DSPSAVF output

```
Library saved  . . . :   CRAIGR                 Release level  . . . :   V5R2M0
 ASP  . . . . . . . . :   1                      Data compressed  . . :   No
 Save file  . . . . . :   CRSAVF                 Objects displayed  . :   6
   Library  . . . . . :     MYLIB                Objects saved  . . . :   9
 Records  . . . . . . :   1880                   Access paths . . . . :   0
 Save command . . . . :   SAVLIB
 Save active  . . . . :   *NO
 Save date/time . . . :   01/13/04   14:55:25

 Type options, press Enter.
   5=Display saved data base file members

 Opt  Object            Type      Attribute     Owner         Size (K)   Data
      CRAIGR            *LIB      PROD          CRAIGR              88   YES
      ADDCODE           *PGM      CLE           CRAIGR              92   YES
      LIBMYDSAPI        *SRVPGM   CLE           QNOTES             220   YES
      ADDCODE           *MODULE   CLE           CRAIGR             188   YES
      MYFILTER          *MODULE   CLE           CRAIGR             196   YES
      MYVLDL            *VLDL                   QNOTES              20   YES  +

 F3=Exit        F12=Cancel
```

Example 8-23   Second page of DSPSAVF

```
Display Saved Objects - Save File

 Library saved  . . . :   CRAIGR                 Release level  . . . :   V5R2M0
 ASP  . . . . . . . . :   1                      Data compressed  . . :   No
 Save file  . . . . . :   CRSAVF                 Objects displayed  . :   9
   Library  . . . . . :     MYLIB                Objects saved  . . . :   9
 Records  . . . . . . :   1880                   Access paths . . . . :   0
 Save command . . . . :   SAVLIB
 Save active  . . . . :   *NO
 Save date/time . . . :   01/13/04   14:55:25

 Type options, press Enter.
   5=Display saved data base file members

 Opt  Object             Type     Attribute     Owner         Size (K)    Data
      ADDTOVLDL          *CMD                   CRAIGR               8    YES
      QCMDSRC            *FILE    PF            CRAIGR              20    YES
      QCSRC              *FILE    PF            CRAIGR             180    YES
```

7. In Example 8-23 notice that the saved library is CRAIGR. When you restore the library you may restore it to an existing library or create a new library. The new library can either be named CRAIGR or you can give it a more descriptive name. We created a library named SSO and ran the following command to restore the files:

   RSTLIB SAVLIB(CRAIGR) DEV(*SAVF) SAVF(MYLIB/CRSAVF) RSTLIB(SSO)

   Note: If you restore to a library other than CRAIGR you will receive an error stating that one object was not restored. The object that does not get restored is the library object. If you create a library by the same name you will not get the error and all the objects will be restored.

   The command to restore to a library named CRAIGR is:

   RSTLIB SAVLIB(CRAIGR) DEV(*SAVF) SAVF(MYLIB/CRSAVF) RSTLIB(*SAVLIB)

8.5.5 Recompilation of the DSAPI exit program on your iSeries

Add library QNOTESAPI to your library list.

The source for the exit program goes into a member in a source physical file -- for instance source member MYFILTER in source physical file QCSRC in library MYLIB.

Create the DSAPI exit program

CRTCMOD MODULE(MYLIB/MYFILTER) SRCFILE(QCSRC/MYLIB) SRCMBR(MYFILTER) DEFINE(OS400) SYSIFCOPT(*IFSIO) DBGVIEW(*ALL)

CRTSRVPGM SRVPGM(MYLIB/LIBMYDSAPI) MODULE(MYLIB/MYFILTER)

CHGOBJOWN OBJ(MYLIB/LIBMYDSAPI) OBJTYPE(*SRVPGM) NEWOWN(QNOTES)

(Note that if you update the service program later, you must end and restart the HTTP task in the Domino server in order for your changes to take effect.)

8.6 Where to find more information

Validation Lists (VLDL) – See descriptions of the create validation list (CRTVLDL) and the delete validation list (DLTVLDL) commands. Also see the QsyFindValidationLstEntry and QsyAddValidationLstEntry APIs.

http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm

Select the appropriate **Geography -> Language (release) -> Programming -> APIs -> APIs by category -> Security -> Validation List APIs**.

- *iAW – iNotes Access for Web*, SC41-5518, and iAW home page in the iSeries Information Center:

  http://publib.boulder.ibm.com/iseries/v5r2/ic2924/books/c4155181.htm

- *Integrating Lotus Domino 6 and WebSphere Express V5 on the IBM @server iSeries Server*, SG24-6998:

  http://www.redbooks.ibm.com/redbooks/SG246998.html

- "Using the Domino HTTP server for iSeries Access for Web: Appendix A" of *iSeries Access for Web V5R2 and WebSphere Host Publisher V4.0*, SG24-6804:

  http://www.redbooks.ibm.com/redbooks/SG246804.html

- Lotus documentation on the Internet at:

  http://www-10.lotus.com/ldd

8.7 Enabling Web Express Logon for WebSphere Host on-Demand

IBM WebSphere Host On-Demand provides cost effective and secure browser-based and non-browser-based host access to users in intranet-based and extranet-based environments. Host On-Demand is installed on a Web server, simplifying administrative management and deployment, and the Host On-Demand applet or application is downloaded to the client browser or workstation, providing user connectivity to critical host applications and data.

Host On-Demand's Web Express Logon feature provides an automated way for users to log on to hosts and host-based applications without having to provide an additional user ID and password. Working in conjunction with EIM and Network Authentication Service, Web Express Logon supports OS/400 (V5R2 and later) telnet-negotiated environments that have Kerberos authentication enabled. It simply extends the existing single sign-on capability of the OS/400 operating system.

For Web Express Logon to function in an OS/400 environment, you must have the following prerequisites in place:

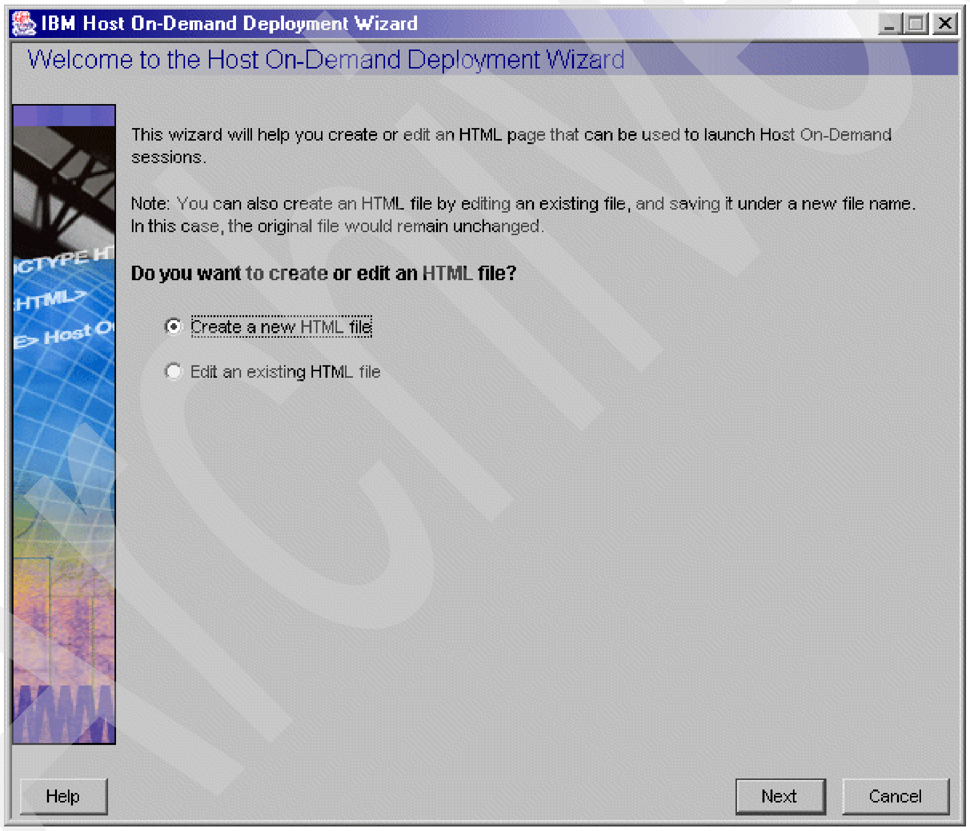
- Host On-Demand Version 8
- Windows Domain Controller (Microsoft Active Directory)
- Key distribution center (KDC)
- Kerberos network authentication enabled on each target OS/400 system
- OS/400 V5R2 or later as the host operating system
- One or more of the following client operating systems:
  - Windows 2000 Professional and Server
  - Windows XP Professional
  - Windows Server 2003

Once you have configured single sign-on in your OS/400 environment, use Host On-Demand's Deployment Wizard to configure Web Express Logon in your single signon environment. The Deployment Wizard allows you to create an HTML file that is used to launch Host On-Demand sessions. Within the Deployment Wizard, you can add, delete, configure, and start sessions. The wizard guides your configuration choices and provides comprehensive help for the features. When you have finished selecting features, it creates the HTML and supporting files for you.

Note: In this scenario, we performed all of the steps all within the Deployment Wizard in one sitting. However, you may decide to create your HTML file first and then configure your session later. Refer to the supplemental notes at the beginning of Step 6 for more information about how to do this.

To begin creating your HTML file, take the following steps:

1. Click **Start -> Programs -> IBM WebSphere Host On-Demand -> Administration -> Deployment Wizard**.
   - If you automatically installed the Deployment Wizard as part of the Windows Host On-Demand server, click **Start -> Programs -> IBM WebSphere Host On-Demand -> Administration -> Deployment Wizard**.
   - If you installed the Deployment Wizard from the Host On-Demand CD-ROM separately, click **Start -> Programs -> IBM WebSphere Host On-Demand Deployment Wizard -> Deployment Wizard**.
2. Select either to create a new HTML file or edit an existing file (Figure 8-39). Click **Next**..


IBM Host On-Demand Deployment Wizard
Welcome to the Host On-Demand Deployment Wizard
Help
Next
Cancel

This wizard will help you create or edit an HTML page that can be used to launch Host On-Demand sessions.

Note: You can also create an HTML file by editing an existing file, and saving it under a new file name. In this case, the original file would remain unchanged.

Do you want to create or edit an HTML file?

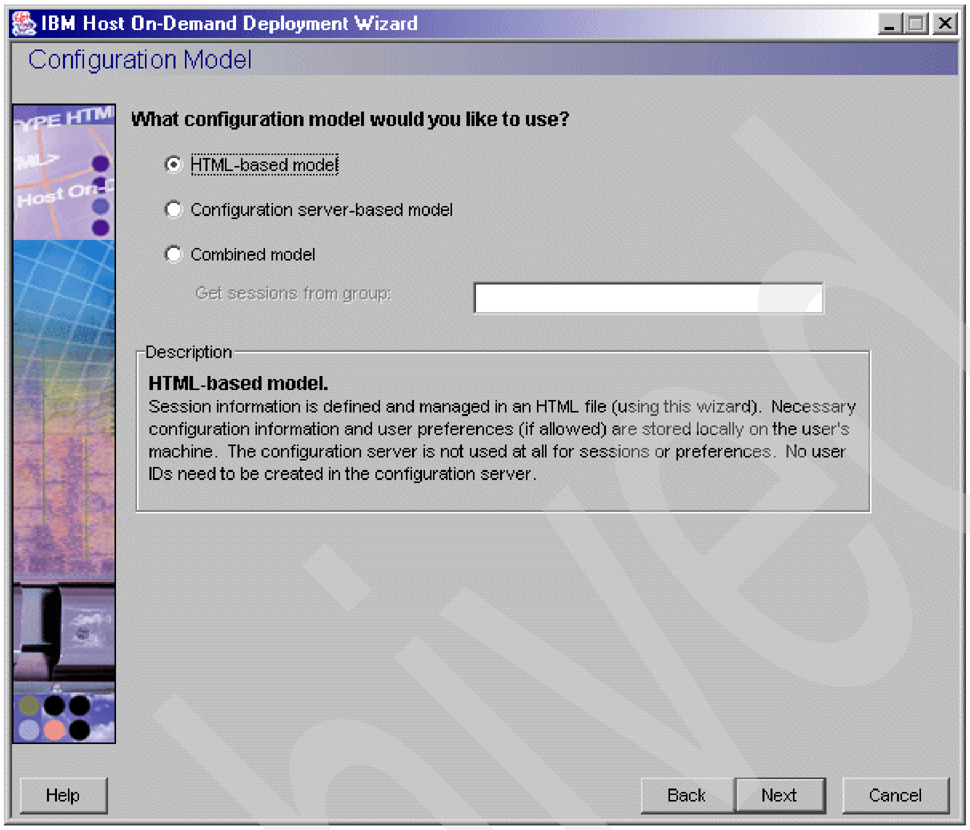
[✓ Create a new HTML file]

Edit an existing HTML file

Figure 8-39 Host on Demand welcome window

3. For the purpose of this scenario we selected the HTML-based configuration model (Figure 8-40). To see the description of each model, click the radio button next to the name of the model. You will have to chose between the three possible types and then click **Next**:
   - HTML-based model.
   - Configuration server-based model. (To select this option you must have already configured a Credential Mapper Server. After providing the address of that server you will go to step 7.)

– Combined model.



What configuration model would you like to use?

[✓ HTML-based model]

Configuration server-based model

Combined model

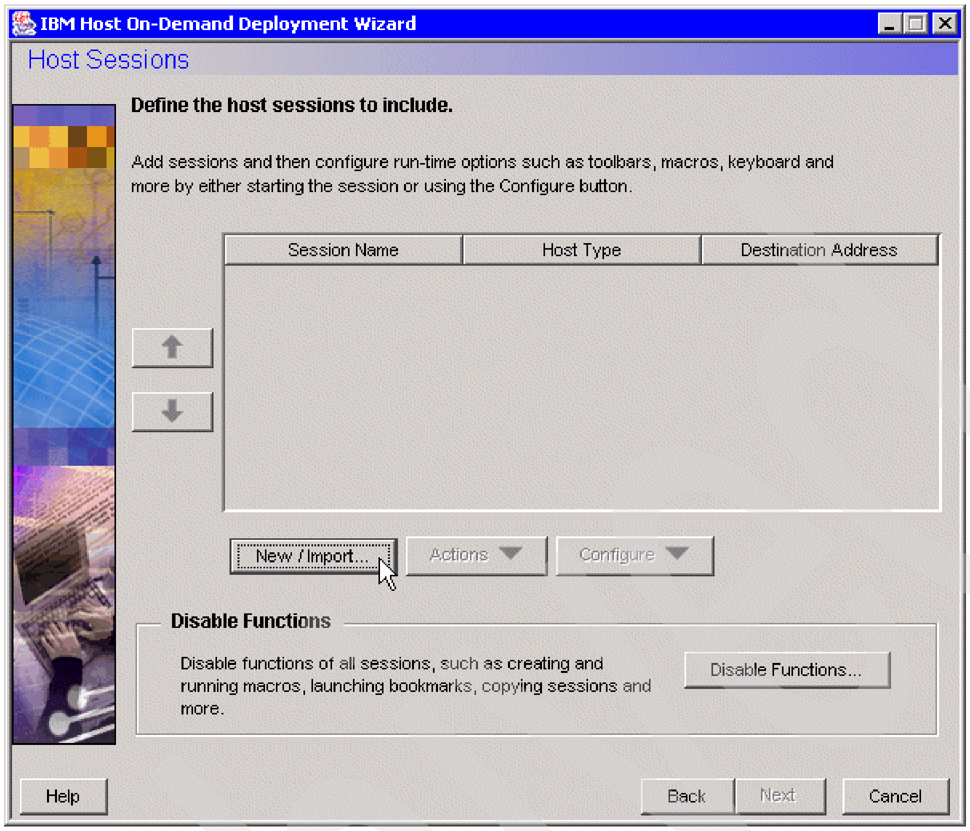
Get sessions from group:

Description

HTML-based model.
Session information is defined and managed in an HTML file (using this wizard). Necessary configuration information and user preferences (if allowed) are stored locally on the user's machine. The configuration server is not used at all for sessions or preferences. No user IDs need to be created in the configuration server.

Figure 8-40 Host on Demand Wizard selection of configuration model

4. The window shown in Figure 8-41 window allows you to either create a new session (default) or import an existing session. On the Host Sessions window, click **New/Import** to open the Add sessions window session.



Define the host sessions to include.

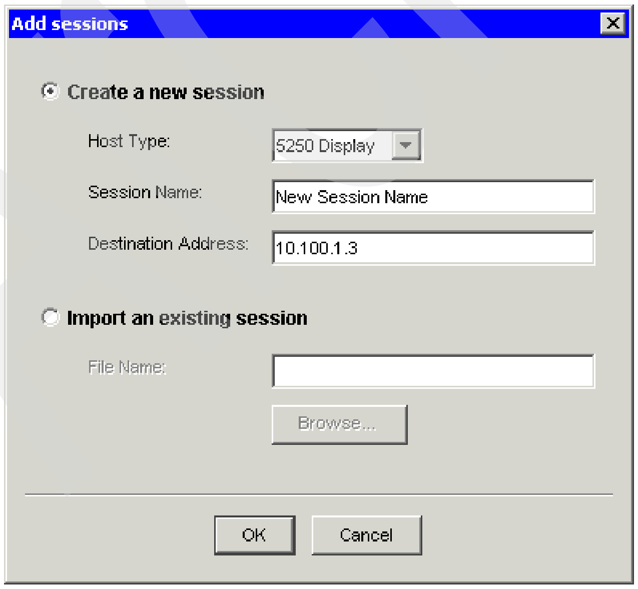
Add sessions and then configure run-time options such as toolbars, macros, keyboard and more by either starting the session or using the Configure button.

Session Name	Host Type	Destination Address

Disable functions of all sessions, such as creating and running macros, launching bookmarks, copying sessions and more.

Figure 8-41 Creating a new host session

5. To create a new session, select a Host Type from the Add sessions window (Figure 8-42), enter a session name, and a destination address. In this scenario, we selected 5250 Display. Click **OK** to return to the Host Sessions window.



(●) Create a new session

Host Type: 5250 Display

Session Name: New Session Name

Destination Address: 10.100.1.3

( ) Import an existing session

File Name:

Figure 8-42 Define the session type

6. In the following step, you will configure your Host On-Demand session to use Web Express Logon.

Note: If you have already created your HTML file and now want to configure it to use Web Express Logon, open the Host On-Demand desktop, right-click the session icon, and select Properties. Skip to Step 6b.

   a. On the Host Sessions window (Figure 8-43), click **Configure -> Properties** to configure your session to use Web Express Logon.

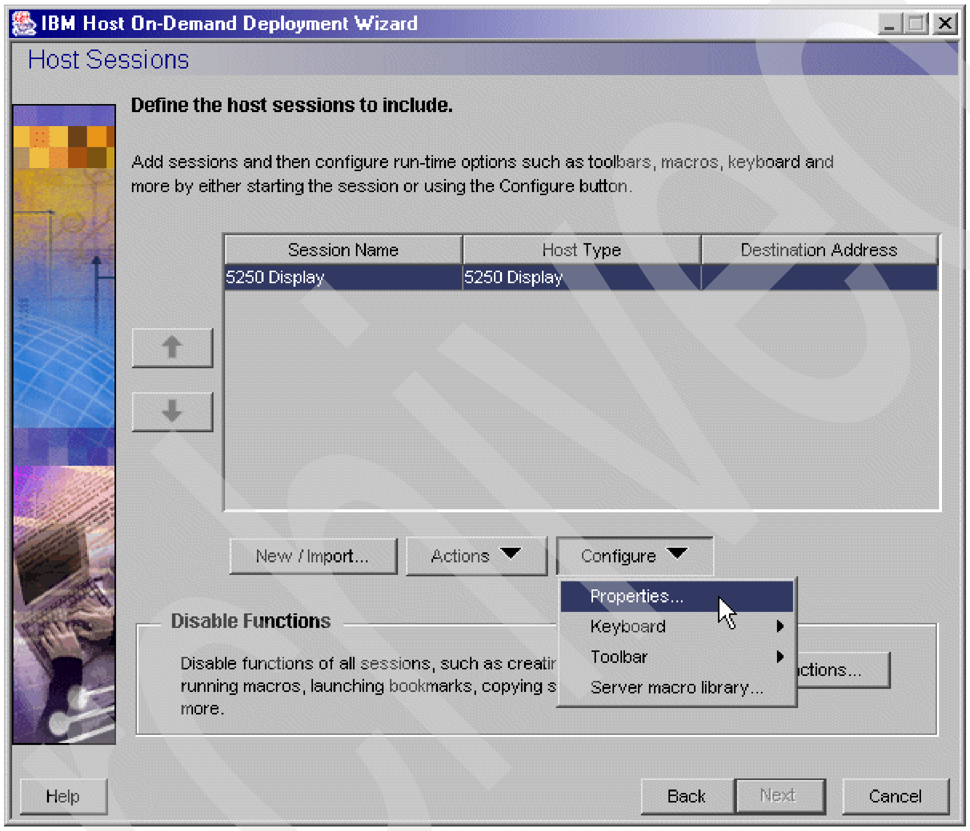
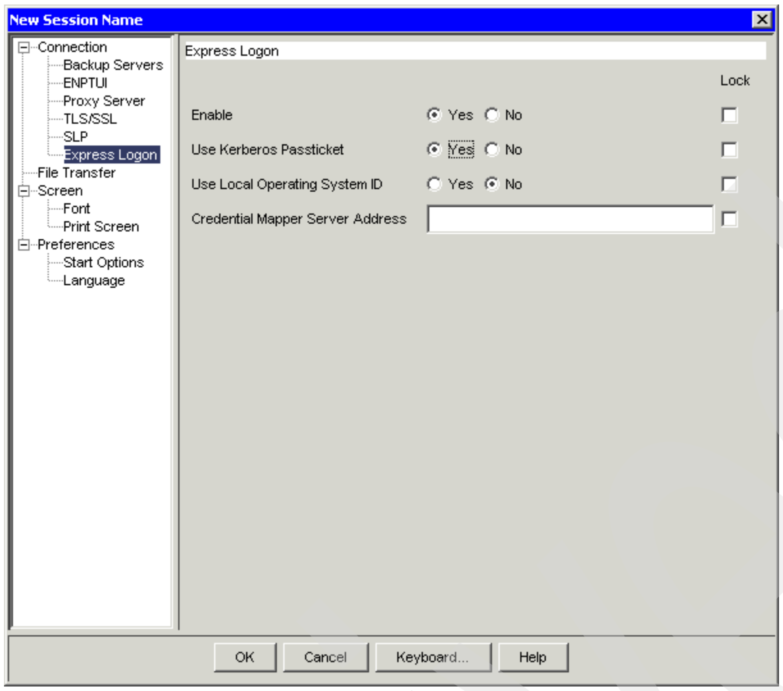

IBM Host On-Demand Deployment Wizard
Host Sessions
Define the host sessions to include.
Add sessions and then configure run-time options such as toolbars, macros, keyboard and more by either starting the session or using the Configure button.
Session Name
Host Type
Destination Address
5250 Display
5250 Display
New / Import...
Actions
Configure
Properties...
Keyboard
Toolbar
Server macro library...
Disable Functions
Disable functions of all sessions, such as creatir running macros, launching bookmarks, copying s more.
ctions...
Help
Back
Next
Cancel

Figure 8-43 Configuring Web Express Logon

   b. Under the Connection option on the left side of the 5250 Display window shown in Figure 8-44, click **Express Logon**. Select **Yes** to enable Express Logon and **Yes** to Use Kerberos Passticket. Accept the default No for Use Local Operating System ID, and leave the Credential Mapper Server Address field blank. You would only need to set these options if you had additional credential challenges that you wanted to automate through a vault-style setup. They do not apply to Kerberos authentication. Once you select to use a Kerberos Passticket, Host On-Demand will be able to retrieve a service ticket from the Windows KDC. This ticket is used to connect to the host system that you identify in the session properties. Click **OK** to return to the Host Sessions window.

   c. When you return to the Host Sessions window, click **Next**..



Express Logon

Lock

Enable [✓ Yes] No ☐

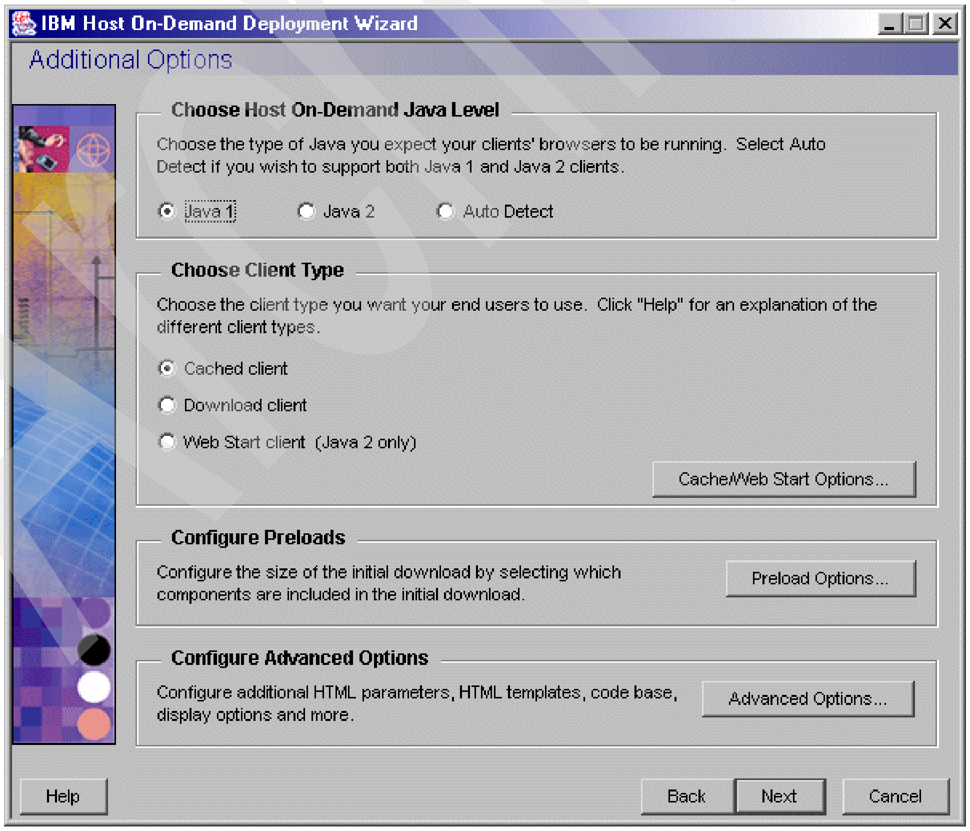
Use Kerberos Passticket [✓ Yes] No ☐

Use Local Operating System ID Yes [✓ No] ☐

Credential Mapper Server Address ☐

Figure 8-44 Enabling Express Logon and Kerberos

7. In the Additional Options window (Figure 8-45) there are no required parameters to set. Review each of the parameters and make any changes that are necessary. Click **Next**.


IBM Host On-Demand Deployment Wizard
Cache/Web Start Options...
Preload Options...
Advanced Options...
Help
Back
Next
Cancel

Additional Options

Choose Host On-Demand Java Level

Choose the type of Java you expect your clients' browsers to be running. Select Auto Detect if you wish to support both Java 1 and Java 2 clients.

[✓ Java 1] Java 2 Auto Detect

Choose Client Type

Choose the client type you want your end users to use. Click "Help" for an explanation of the different client types.

[✓ Cached client]
Download client
Web Start client (Java 2 only)

Configure Preloads

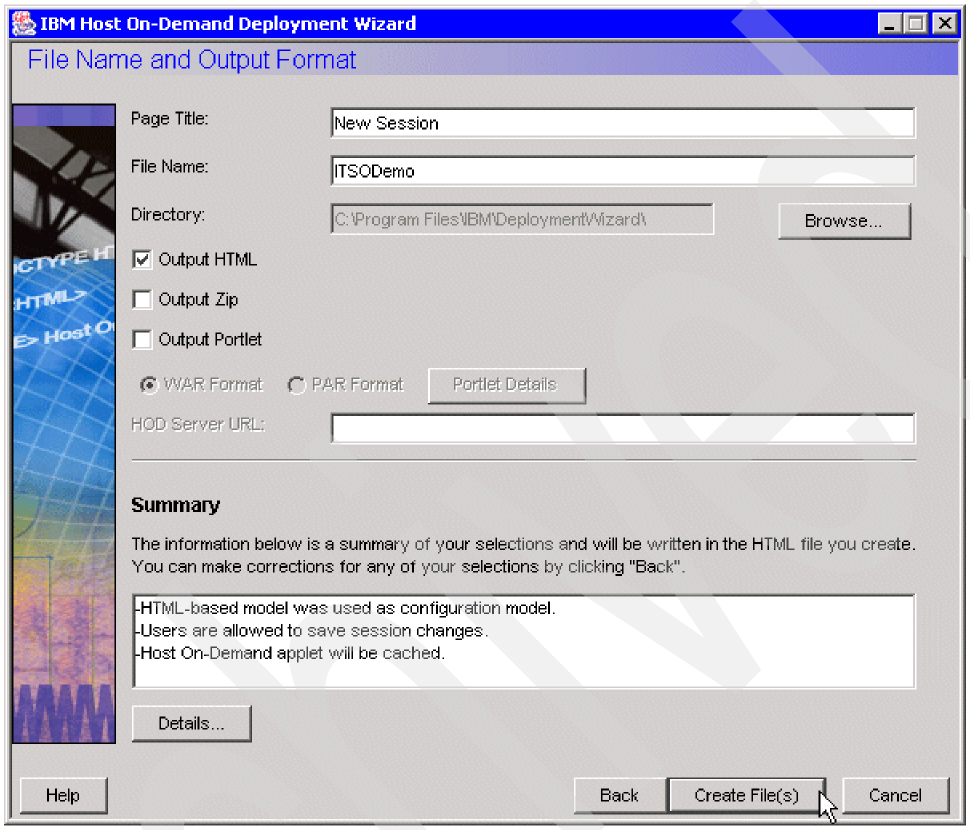
Configure the size of the initial download by selecting which components are included in the initial download.

Configure Advanced Options

Configure additional HTML parameters, HTML templates, code base, display options and more.

Figure 8-45 Additional options window

8. In the File Name and Output Format window (Figure 8-46), enter the page title, the file name, and choose the directory where you want to save your file. You should save it to the Host On-Demand server in a directory known to your Web server; usually, this directory is your Host On-Demand server's publish Directory. Click **Create File(s)** to finish creating your HTML file.



Page Title: New Session

File Name: ITSODemo

Directory: C:\Program Files\IBM\DeploymentWizard\

[✓] Output HTML

[ ] Output Zip

[ ] Output Portlet

HOD Server URL:

Summary

The information below is a summary of your selections and will be written in the HTML file you create. You can make corrections for any of your selections by clicking "Back".

-HTML-based model was used as configuration model.
-Users are allowed to save session changes.
-Host On-Demand applet will be cached.

Figure 8-46 Create the HTML file

After creating the HTML file(s), you have completed enabling Host on Demand to take advantage of you single signon environment. For more information about Version 8 of WebSphere Host On-Demand, visit the Host On-Demand Information Center:

`http://www.ibm.com/software/webservers/hostondemand/library/v8infocenter/hod/en/help/2tabcontents.html`

For a complete discussion of Web Express Logon for Host On-Demand, including troubleshooting tips, refer to the following white paper:

`http://www.ibm.com/software/network/library/whitepapers/wel.pdf`

Host On-Demand also offers a Web-based tutorial to help you configure Web Express Logon:

`http://www.ibm.com/software/webservers/hostondemand/library/v8infocenter/hod/en/tutorials/webexpress/purpose.html`

9

Programming APIs and examples

Enterprise Identity Mapping (EIM) provides the mechanics for cross-platform user identity management. You can write applications that use EIM APIs to perform look up operations for user identities within an enterprise. Each of these APIs are supported across IBM @server platforms. Within the C API description, implementation notes have been added to indicate the functional IBM @server differences across each of these operating systems: AIX, Linux, OS/400, Windows, and z/OS.

This chapter provides an overview of the different programming APIs that are available for interfacing with the EIM domain and the information contained therein. We start with a description of the Java and C APIs. For Java, we discuss the different classes and interfaces that are available, concentrating on some of the commonly used methods to create and retrieve data. With the C APIs, we concentrate on a code example which uses exit points in iSeries Access for the Web and Domino Web Access to provide single signon functionality.

After the discussion of each programming API we give some code examples to show the practical uses of these APIs in relation to how they could be incorporated to fit the needs of businesses and developers.

At the end of the chapter there is a description of an EIM demo program that can be downloaded. The program demonstrates how the EIM APIs can be utilized.

9.1 Java EIM API

The Java EIM API comes as a jar file and can be found here:

http://publib.boulder.ibm.com/eserver

Select the appropriate **Geography -> Language -> Enterprise Identity Mapping -> Server-specific EIM information -> EIM Availability and Series details**.

The Javadocs for this API can be downloaded with the jar file at the same location.

We also see, in this chapter, that there is security in the APIs so that unauthorized users cannot log in and delete the EIM domain. This security information is enforced in LDAP so that no specific security information needs to be applied into the APIs themselves.

9.2 Java classes and interfaces

This section contains an explanation of the different Java classes that are contained within the EIM jar file. The Java EIM APIs interface with an LDAP server. To do this the EIM APIs make use of existing Java classes which the classes call to perform functions on behalf of EIM. Why should the EIM APIs interface directly with LDAP when there are already functionality to perform these operations? It is worth noting that the LDAP classes should *not* be used directly to access EIM information in LDAP, this could lead to an EIM domain in an organization becoming unusable. The consequences could be catastrophic for a large EIM domain if it becomes unusable. We strongly recommend only interfacing with EIM through the EIM jar. For more information on the Java Naming and Directory Interface (JNDI) classes see:

http://java.sun.com/products/jndi/

9.2.1 DomainManager class

The DomainManager class is the starting class for all interactions with EIM. Through this class all EIM operation can be performed. The DomainManager class cannot be instantiated directly, rather the method getInstance is called to return a DomainManager object which can then be used. The methods contained within the DomainManager class allow a developer to access domains and create domains. It is possible to retrieve an individual domain or a list of all the EIM domains that are available on the server, which hosts an EIM domain.

Important: It is extremely important to note that EIM needs to be configured on the iSeries, or your IBM @server, before a domain can be created. In the configuration of EIM a root node can be created in LDAP to store this EIM information. EIM APIs cannot create a root node in LDAP, if a new EIM domain wants to be created, the administrator of the LDAP server must do this. However, if in the EIM configuration, a subtree was created for different EIM domains, then the createDomain method will be able to create this tree, because the parent node has already been created.

9.2.2 The java.util.Set class

In this DomainManager class and subsequent classes, when multiple objects such as domain lists and registry lists, for example, are returned, they are an object of the java.util.Set class. This class is a unique list that contains no duplicates. Because the list does not allow duplicates there cannot be multiple null entries. A set cannot contain different objects which hold identical information. For example, object1.equals(object2). A set also cannot contain itself. If mutable objects are used in a set, then the resulting behavior is not defined if an object is changed so that it equals another object.

9.2.3 Domain class

After the DomainManager object returns a domain object either through the createDomain method or getDomain(s) methods it can be used to directly interface with everything inside the domain. The domain class is the class in EIM which holds the most functionality. Actions which can be performed are:

- Access information about the domain itself.
- Access information about the system the domain resides on.
- Registries can be created and accessed.
- EIM Identifiers can be created and accessed.
- Retrieve associations for one to many source and target associations.
- Security information about the domain can be accessed.
- Information about the different authorities can be accessed.

9.2.4 Registry interface

In the domain object there are methods for retrieving registries, these are the getRegistries methods which take various parameters. All getRegistries methods return a set object holding the list of registries. There is no way of individually specifying a registry to return however it is possible to specify parameters to narrow down the number of registries returned. Although even using all available parameters it is possible that duplicates will occur if registries have the same configuration information.

The set object returned holds a list of registry objects although this list may only hold one object. The registry object returned has many methods most of which deal with retrieving information about the registry rather than changing information stored within it. The only information that can be changed from the registry object is the description associated with the registry and its aliases. A lot of valuable information can be retrieved from a registry object such as:

- A list of users that pertain to this registry
- The type of this registry
- The registries description
- The name of the registry
- A list of aliases that refer to this registry

Using this object the registry can be deleted from the LDAP directory.

9.2.5 SystemRegistry interface

The registry interface above provides methods which are available to the different types of registry available to us. There are two more registry interfaces which extend this registry functionality these are system registries and application registries. The application registry is explained below.

As the system registry interface extends the registry object it allows access to all the properties and methods that are contained therein. As defined in Chapter 6 system registries are deemed to be registries which have there own user base. A good example of this would be a Domino server where the Notes and Domino environment has its own users through the use of ID files.

9.2.6 ApplicationRegistry interface

As defined in 6.3.4, “EIM registry definitions” on page 78, application registries are registries whose users relate to the users of a system that this application resides on. An example of this would be an RPG application whose users are OS400 users but who use this application. When applications share system user names like this it is a good example to use an application registry. It is however user preference if this is done or not. As application registries reside on system registries a system registry needs to be referenced in the addApplicationRegistry method of the Domain class.

There is only one new method that appears in this new extended interface. This method is getSystemRegistryName() and returns the name of the system registry where this application registry resides.

9.2.7 RegistryAlias class

The RegistryAlias class is used in the Domain class methods getRegistryByAlias and getRegistryNames. These methods return a java.util.Set object which holds a list of all registries which have a registry alias of the type specified. The registry alias object is created using 2 parameters. The first is the value held by the alias, this is called the name. The second is a reference to this value, this is called the type. An example of this would be:

```
RegistryAlias regAlias = new RegistryAlias(“127.0.0.1”,”TCP/IP Address”);
```

Information about the RegistryAlias object can be retrieved from two methods getName and getType which refer to the parameters listed above. Registry objects are immutable, they can’t be changed after instantiation, for this reason the Registry methods addAlias and deleteAlias are used to append and delete alias definitions to registries. As registry aliases are defined in the Registry Interface both application and system registries are able to use RegistryAlias objects.

9.2.8 Eid interface

The Eid interface represents the EIM Identifier. This EIM Identifier represents an entity in EIM, this can relate to a person or a service. The Eid interface allows a lot of functionality to interact with EIM Identifiers. The operations that can be performed are:

- Associations can be added and removed for registries.
- Aliases can be added and removed for this EIM Identifier.
- Additional Information can be added and removed for this EIM Identifier.
- Lookup operations for target registries can be retrieved.
- Information about the EIM Identifier such as the name, description can be retrieved.
- The Eid can also be deleted.

Eid objects are returned in a java.util.Set object from methods in the Domain class. These methods are used to query the EIM domain for a set of Eids based upon:

- User aliases
- The name of the EIM Identifier
- EIM Identifiers which represent the same universal unique identifier

From the domain object there is no way of retrieving a set of Eid’s based on a registry parameter. We see how to do this in the RegistryUser Interface below.

9.2.9 RegistryUser interface

To retrieve a list of users which pertain to a specific registry we use the getUsers methods of the Registry Interface. The getUsers methods return a set of RegistryUser objects. Set's of UserRegistry objects can also be returned from the Domain class methods used for retrieving associations. The RegistryUser interface provides methods in order to retrieve:

- Target association information used by applications to authorize a users actions
- Retrieve the Eid corresponding to this user

The RegistryUser Interface is used to retrieve a list of all users in the Registry and to find target associations between users source and target identifiers.

9.2.10 ConnectInfo class

The ConnectInfo class provides an object which is used for binding to LDAP. This bind occurs when domain operations are performed in the DomainManager class. This class allows many different ways of connecting to LDAP through:

- User name and password
- Using different encryption options with a user name and password
- Using an SSLInfo object as described below.
- Using credentials and SSL Information

It is possible when using credentials to connect to the LDAP server. Using Kerberos as the credential if SSL is not used the value of null is used instead. For a list of all credential types, consult the Javadocs.

9.2.11 SSLInfo class

This is the class that is used in the ConnectInfo constructor if an SSL connection to LDAP is to be created. After creation of the SSLInfo object through the constructor you cannot set the SSLInfo parameters, it is advisable to specify all SSLInfo information needed in the constructor. The methods that are available to the SSLInfo class are only get methods to retrieve the information that is set in the constructor.

9.2.12 AccessContext interface

The AccessContext interface provides access to EIM authorization management functions. An AccessContext object is returned from the Domain class method getAccessContext. This class is used to provide an interface which allows users to be given different administrative rights in the database. It also allows for users to be added and deleted from these administrative groups. These different administrative groups determine what methods a user is able to call in the EIM APIs. For a list of these different authorities, consult 6.3.7, "EIM authorities" on page 85.

9.2.13 UserAccess class

The UserAccess interface holds methods which represent an individual users authority in the EIM domain. An UserAccess class can either be instantiated directly or obtained through the UserAccess object is to call the getUserAccess method of the AccessContext Interface. Constructor parameters determine the information associated with this users level of access that this user has in the system although this cannot override the privileges that this user actually has.

9.2.14 EIMException class

This is the class used to define that an error occurred while using the EIM classes. The EIMException class overrides some of the methods inherited from the Exception class and sub-classes. It also builds on this exception class by incorporating reason codes to represent the base cause of this exception.

For more details on the EIM Java classes and their methods, consult the EIM Javadocs.

9.3 Security in the Java classes

In this section we look at the EIM authorities who have access to certain methods and classes within the EIM Java APIs. The LDAP administrator is not listed in the authorities specified below but this user has access to all EIM methods. The security here although it applies to the Java methods and objects listed below is not specified in the Java class, or the C APIs, the security for the EIM authorities is enforced in LDAP. This abstractions allows the creation of EIM APIs to be much simpler without having to code in specific security information into each API. This could lead to unforeseen security flaws in each API rather than a central place to control security.

9.3.1 DomainManager class

Table 9-1 lists the methods in the DomainManager class and who has the authority to access them.

Table 9-1 DomainManager class authority authorization

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
createDomain	X				
getDomain	X	X	X	X	X
getDomains	X	X	X	X	X
getInstance	X	X	X	X	X

9.3.2 Domain class

Table 9-2 lists the methods in the Domain class and who has the authority to access them.

Table 9-2 Domain class authority authorization

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
addApplicationRegistry	X				
addEid	X		X		
addSystemRegistry	X				
addUniqueEid	X		X		
delete	X				

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
disconnect	Even users without an authority can access this method				
findTargetFromSource	X	X	X	X	X
getAccessContext	Even users without an authority can access this method				
getDescription	Even users without an authority can access this method				
getDn	Even users without an authority can access this method				
getEids	X	X	X	X	X
getEidsByAlias	X	X	X	X	X
getEidsByName	X	X	X	X	X
getEidsByUuid	X	X	X	X	X
getHost	Even users without an authority can access this method				
getName	Even users without an authority can access this method				
getPort	Even users without an authority can access this method				
getRegistries	X	X	X	X	X
getRegistriesByAlias	X	X	X	X	X
getRegistryNames	Even users without an authority can access this method				
getUrl	Even users without an authority can access this method				
isConnected	Even users without an authority can access this method				
isSSL	Even users without an authority can access this method				
setDescription	Even users without an authority can access this method				

9.3.3 Registry interface

Table 9-3 lists the methods in the Registry interface and who has the authority to access them.

Table 9-3 Registry Interface authority authorization

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
addAlias	X	X			X
delete	X				
getAliases	X	X	X	X	X
getDescription	X	X	X	X	X
getKind	X	X	X	X	X
getName	Even users without an authority can access this method				

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
getType	X	X	X	X	X
getUsers	X	X	X	X	X
getUuid	X	X	X	X	X
removeAlias	X	X			X
setDescription	X	X			X

9.3.4 Eid class

Table 9-4 lists the methods in the Eid class which have authority associated with them.

Table 9-4 Domain class authority authorization

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
addAdditionalInfo	X		X		
addAlias	X		X		
addAssociation (For source and admin)	X		X		
addAssociation (For target)	X	X			X
delete	Even users without an authority can access this method				
findTarget	X	X	X	X	X
getAdditionalInfo	Even users without an authority can access this method				
getAliases	Even users without an authority can access this method				
getAssociations	X	X	X	X	X
getDescription	Even users without an authority can access this method				
getName	Even users without an authority can access this method				
getUuid	Even users without an authority can access this method				
removeAdditionalInfo	X		X		
removeAlias	X		X		
removeAssociation (For source and admin)	X		X		
removeAssociation (For target)	X	X			X
setDescription	X		X		

9.3.5 RegistryUser class

Table 9-5 lists the methods in the RegistryUser class which have authority associated with them.

Table 9-5 RegistryUser class authority authorization

Method / Authority	EIM Administrator	EIM Registries Administrator	EIM Identifiers Administrator	EIM Mapping Lookup	EIM authority to individual registries
addAdditionalInfo	Even users without an authority can access this method				
getAdditionalInfo	Even users without an authority can access this method				
getAssociatedEids	X	X	X	X	X
getDescription	Even users without an authority can access this method				
getRegistryName	Even users without an authority can access this method				
getTargetUserName	Even users without an authority can access this method				
removeAdditionalInfo	Even users without an authority can access this method				
setDescription	Even users without an authority can access this method				

As you can see from the tables there are many commands which only users with a predetermined level of access will be able to use. This restricts the functionality which can be used when performing mapping lookups for average users. It is however important to note that is users are only performing mapping lookups then they do not need a high level of access.

9.4 Java example: ReportEIM

In this section we explain some code examples to help you understand the EIM Java classes. How the different hierarchy of objects fit into the EIM classes and what operations can be performed using the classes. There are no classes used in these code examples other than the classes and interfaces explained above in 9.2, "Java classes and interfaces" on page 188.

Many of the methods used in this section have similar names to the actual methods in the EIM jar. The methods listed here are not to be confused with these methods. I have written my own methods as a wrapper to these methods to allow for a centralized place for error trapping and program specific operations.

9.4.1 Constants

The example below shows some of the information that is used in the following methods when domains are accessed and created. We have included this information as variables at the start of the `ReportEIM` class however a lot of this information could be incorporated into a constructor and assigned to variables when the `ReportEIM` class is created.

Example 9-1 Constants

```
   //Constants for creating and accessing domains
   private String user = "cn=administrator";
outputtedprivate String password = "ldappw";
   private ConnectInfo connectInfo = new ConnectInfo(user, password);
   private String system = "ldap://as20.itsoroch.ibm.com:389/";
   private String domain = "ibm-eimdomainname=BlueNotesEIMDomain";
   private String ldapInfo = system + domain;
   private String domaindesc = "Created By Blue Notes EIM Administrator (BlueNotes.com)";

   //These constants are used to determine which report to run
   public static final int ReportOne = 1;
   public static final int ReportTwo = 2;
   public static final int ReportThree = 3;
   public static final int ReportFour = 4;
   public static final int DeleteDomain = 5;
```

9.4.2 The createAssociationTypeMap method

Constants exist in the Association interface. These constants are used when adding associations to an EIM Identifier. These constants are integer values that are referenced through accessing the properties of the Association interface. When using methods in the Association interface such as getAssociationType() this integer value is returned. This integer is not very descriptive and so I have created a HashMap object to hold these values with a related description of the integer.

Example 9-2 The createAssociationTypeHashMap method

```
Hashmap h;

private void createAssociationTypeHashMap()
{
   h = new HashMap();
   h.put(new Integer(Association.EIM_ADMIN), "admin");
   h.put(new Integer(Association.EIM_ALL_ASSOC), "all");
   h.put(new Integer(Association.EIM_SOURCE), "source");
   h.put(new Integer(Association.EIM_SOURCE_AND_TARGET),"source and target");
   h.put(new Integer(Association.EIM_TARGET), "target");
}
```

9.4.3 The createRegistryTypeHashMap method

For the same reasons as above the registry type definitions in EIM are also undescriptive as to what they represent. To combat this I have created another HashMap object which holds the registry types and an associated description.

Example 9-3 The createRegistryTypeHashMap method

```
Hashmap rh;

private void createRegistryTypeHashMap()
{
   rh = new HashMap();
   rh.put(Registry.EIM_REGTYPE_AIX, "AIX");
   rh.put(Registry.EIM_REGTYPE_KERBEROS_EX, "Kerberos EX");
   rh.put(Registry.EIM_REGTYPE_KERBEROS_IG, "Kerberos IG");
   rh.put(Registry.EIM_REGTYPE_LDAP, "LDAP");
   rh.put(Registry.EIM_REGTYPE_NDS, "Novell Directory Services");
   rh.put(Registry.EIM_REGTYPE_OS400, "OS400");
   rh.put(Registry.EIM_REGTYPE_POLICY_DIRECTOR, "Policy Director");
   rh.put(Registry.EIM_REGTYPE_RACF, "RACF");
   rh.put(Registry.EIM_REGTYPE_WIN2K, "WIN2K");
   rh.put(Registry.EIM_REGTYPE_X509, "X509 Certificate");
   rh.put(new Integer(Registry.EIM_ALL_REGISTRIES), "All Registries");
```

```
    rh.put(new Integer(Registry.EIM_APPLICATION_REGISTRY), "Application Registry");
    rh.put(new Integer(Registry.EIM_SYSTEM_REGISTRY), "System Registry");
}
```

9.4.4 The getDomain method

This method is a wrapper for the `Domain` class method getDomain. I have created a wrapper for this method as much of the information used within the `Domain` is reused each time a new domain is accessed. It may not be common for an organization to have more than one EIM domain but it is a possibility.

An example of this is if an organization has multiple Key Distribution Centers (KDCs) for Kerberos. These KDCs typically relate to a specific domain within an organization and it would be a logical place to also hold an EIM domain to represent the people and servers held within this domain. In this example, we say the KDCs are on different networks. Using Kerberos cross realm trust tickets could be passed between realms enabling a user to use a different EIM domain on that network. The user performing this action would also need an EIM Identifier in this other EIM domain and an association for the server/service that is to be accessed.

This method takes a `domainName` variable and a boolean `create` variable. The domainName variable is combined with the system constant listed above to create an LDAP url to this domains information.

The boolean value is used to determine that if there is an exception in getting the domain, for example if it does not exist, does a new domain want to be created. The boolean value in this method and subsequent getRegistry and getEids methods are used to provide an example of how to add information into an EIM domain. These booleans purely provide a way of populating a domain with sample data, this is not a best practice by any means and also allows us a fitting way of showing you how objects can be created as well as accessed.

If the domain trying to be accessed does not exist then the following exception is thrown:

```
EIM exception text=com.ibm.eim.jndi.DomainJNDI:connect: failed to connect to initial
directory context. Root exception class=class javax.naming.NameNotFoundException. Root
exception text=[LDAP: error code 32 - No Such Object].
```

In this example the exception is caught and a domain will be created, again this is only to create a domain with test data if no domain exists.

Example 9-4 The getDomain method

```
private Domain getDomain(String domainName, boolean create)
{
    System.out.println("Getting domain ...");
    try
    {
        return dm.getDomain(system + domainName, connectInfo);
    }
    catch (EimException e)
    {
        System.out.println("Unable to get existing domain");
            System.out.println(e.toString());
            try{return (create ? createDomain() : null);}
            catch(EimException ex){ex.printStackTrace();}
      }
    return null;
}
```

9.4.5 The getAllDomains method

Here is an example of a method which returns multiple objects. As described in “The java.util.Set class” on page 188 when multiple objects are returned from a method in the EIM classes they are always returned as a java.util.`Set` object. The return type for this method is hence a `java.util.Set` object. It however does not provide a list of Domain objects rather it provides a list of strings representing the available domains in the LDAP server. This is also why I have written the wrapper for the getDomain method to take in the domain name as a string and concatenated it with the LDAP base URL in the dm.getDomain method to make the ability of getting different domains more simple.

Example 9-5 The getAllDomains method

```
private Set getAllDomains() throws EimException
   {
      System.out.println("Getting all domains ...");
      try
      {
         Set s = dm.getDomains(system, connectInfo);
         return s;
      }
      catch (EimException e)
      {
         System.out.println("Unable to get domains");
            System.out.println(e.toString());
            return null;
        }
    }
```

9.4.6 The createDomain method

The createDomain method uses information in the constants part of the class to create a domain. Although the domain is created, if the domain used by the iSeries for EIM is different to this domain created here then the EIM configuration will have to be corrected to use this domain. See 8.3.3, “Adding the iSeries server to the EIM domain” on page 146. This example details adding a second iSeries to an existing EIM domain however the steps to follow are identical, however you need to reference the local iSeries instead of the iSeries containing the domain to connect with.

Example 9-6 The createDomain method

```
   private Domain createDomain() throws EimException
   {
      System.out.println("Creating new domain");
      try
      {
         d = dm.createDomain(ldapInfo, connectInfo, domaindesc);
         return d;
      }
      catch (EimException e)
      {
         System.out.println("Unable to create domain");
              //throw e;
              return null;
      }
   }
```

9.4.7 The getRegistries method

In this code example we look at the getRegistries method of the ReportEIM class. This method is another wrapper which I have written to allow error trapping to be performed and segregated from the program methods which call this method. This makes the reporting methods much simpler to implement and cleaner to follow. Unlike the getDomains method the DomainManager getRegistries method returns a jav.util.Set of Registry objects. There are other getRegistry methods in the DomainManager class which take in parameters to access a specific registries. In this reporting example we are only using the getRegistries method to retrieve all registries from a specified domain.

Important: It is very important to note that when a list of registries is returned in a Set object the objects contained inside are in no defined order and are probably not in the order which you added the registries to the EIM domain. You should not rely on this order in any code to determine what actions to perform on a particular item in the list.

Example 9-7 The getRegistries method

```
private Set getRegistries(Domain d, boolean create) throws EimException
{
   Set s = d.getRegistries();
   try
   {
      if(!s.isEmpty())
         return s;
      else
         return (create ? createRegistries(d) : s);
    }
   catch (EimException e)
   {
      System.out.println("Unable to get registry.");
      e.printStackTrace();
      return s;
   }
}
```

9.4.8 The createRegistries method

This method shows you how to create registries using the domain methods addSystemRegistry and addApplicationRegistry. This method also shows you how to add aliases to your registries. Currently in this release of EIM there are six pre-defined registry alias types that can be used. The drop down box that allows you to select these six pre-defined types is editable so you can defined your own alias types to be included in the alias list in iSeries Access.

Attention: If your iSeries Access is not at service level SI09809 this can cause problems. In previous iSeries Access service levels when a user defined alias type is used it will appear as that type in the list. However, if the registry properties is closed as re-opened the type for all user defined alias types will be changed to *TCP/IP address*. This is only a cosmetic difference to the information displayed in the list not to the actual data itself. If you were to output the aliases for a registry using the APIs then the values would be the same as when they were entered by the user.

The six different pre-defined types and a brief description are listed in Table 9-6.

Table 9-6 Alias type definitions

Plain Text	Java Doc Property	Description
DNS Host Name	RegistryAlias.TYPE_DNS	Constant for a DNS host name registry alias
Issuer distinguished name	RegistryAlias.TYPE_ISSUER	Constant for a issuer distinguished name registry alias type
Kerberos realm	RegistryAlias.TYPE_KERBEROS	Constant for a Kerberos realm registry alias
LDAP DNS host name	RegistryAlias.TYPE_LDAPDNSHOSTNAME	Constant for a LDAP DNS host name registry alias type
Root distinguished name	RegistryAlias.TYPE_ROOT	Constant for a root distinguished name registry alias type
TCP/IP address	RegistryAlias.TYPE_TCPIP	Constant for a TCPIP address registry alias type
Other	RegistryAlias.TYPE_OTHER	Constant for an other registry alias type

This last property Other does not appear in service level SI09809 however it is in the implementation of the V5R3 iSeries Access program. The current iSeries Access for V5R3 is available as a beta download from the IBM Web site. This can be located on the Internet at:

http://www.ibm.com/servers/eserver/iseries/clientaccess/

It is worth noting that at the time of writing this book this beta is incomplete and we would not recommend installing it on a critical system to use every day rather for a look at certain features and then to remove it.

Example 9-8 The createRegistries method

```
//Registries To Add
private String sysregistry = "My_iSeries";
private String sysregistrytype = Registry.EIM_REGTYPE_OS400;
private int sysregistrykind = Registry.EIM_SYSTEM_REGISTRY;

private String appregistry = "LDAP";
private String appregistrytype = Registry.EIM_REGTYPE_LDAP;
private String appregistrydesc = "My LDAP Server";

private String sysregistry2 = "My_AIX";
private String sysregistrytype2 = Registry.EIM_REGTYPE_AIX;

private String sysregistry3 = "My_RACF";
private String sysregistrytype3 = Registry.EIM_REGTYPE_RACF;

private Set createRegistries(Domain d) throws EimException
{
   try
   {
      System.out.println("Adding registries.");
      Registry reg;
      reg = d.addSystemRegistry(sysregistry, sysregistrytype, "iSeries Production Server", null);
      reg.addAlias(new RegistryAlias("91.92.94.94","UserDefinedInAPIs"));
      reg.addAlias(new RegistryAlias("host2.domain.com","DNSHostName'"));
```

```
         reg = d.addApplicationRegistry(appregistry, appregistrytype, appregistrydesc, sysregistry);
         reg.addAlias(new RegistryAlias("91.92.94.95","UserDefinedInAPIs"));
         reg.addAlias(new RegistryAlias("host.domain.com","DNSHostName"));
         reg = d.addSystemRegistry(sysregistry2, sysregistrytype2, "AIX Box", null);
         reg.addAlias(new RegistryAlias("91.92.94.96","UserDefinedInAPIs"));
         reg = d.addSystemRegistry(sysregistry3, sysregistrytype3, "RACF Box", null);
         reg.addAlias(new RegistryAlias("91.92.94.97","UserDefinedInAPIs"));
         return d.getRegistries();
      }
      catch (EimException e)
      {
         System.out.println("Unable to add registry.");
         e.printStackTrace();
      }
      return null;
   }
```

In the code example above you will see that the addSystemRegistry method uses a parameter which is null. This parameter is the *Universal Unique Identifier* (Uui) for the registry.

9.4.9 The getEids method

This code example shows how to return a list of EIM Identifiers from a domain. Again we see the benefit of using a wrapper class to trap errors and we can define what is returned to our program. The boolean `create` is also passed to indicate that if there is no EIM Identifiers returned then we can create some and return them. This again is purely a way of populating some demo information into EIM so that something can be outputted which we see subsequently.

Example 9-9 The getEids method

```
   private Set getEids(Domain d, boolean create) throws EimException
   {
      Set s = d.getEids();
      try
      {
         if(!s.isEmpty())
            return s;
         else
            return (create ? createEids(d) : s);
      }
      catch (EimException e)
      {
         System.out.println("Unable to get eid.");
         e.printStackTrace();
      }
      return s;
   }
```

9.4.10 The createEids method

This is the method that is called from the above getEids method when the boolean flag is set to true and there are no EIM Identifiers listed in the domain. This code example shows how to add an Eid object to the domain and how to add association, additional information and aliases. It also gives you an example of what additional information may be used for. By using additional information with a special character, in this case a colon, it would be possible for us as developers to substring up to this special character. This would leave us with two parts a label and a value. We could test the label to see what sort of operation to perform. An example of this could be that if an error occurred or an EIM exception when trying to retrieve a target identity this value could be used to e-mail the user detailing the problem so that the user could forward this information onto the system administrator.

Example 9-10   The createEids method

```
//People To Add
private String eidname1 = "Craig Wayman";
private String eiddesc1 = "EIM Code Author";
private String eidname2 = "Charlie Brown";
private String eiddesc2 = "Loves baseball";
private String eidname3 = "Eric";
private String eiddesc3 = "When Eric eats a banana..";
private String eidname4 = "Clark Kent";
private String eiddesc4 = "Better than Eric?";

private Set createEids(Domain d) throws EimException
{
   try
   {
      System.out.println("Adding eid ..");
      Eid eid;
      eid = d.addEid(eidname1,eiddesc1);
         eid.addAssociation(Association.EIM_SOURCE,"My_iSeries","WaymanC");
         eid.addAssociation(Association.EIM_TARGET,"LDAP","craig");
         eid.addAssociation(Association.EIM_ADMIN,"My_AIX","namyawC");
         eid.addAssociation(Association.EIM_TARGET,"My_RACF","graic");
         eid.addAdditionalInfo("emailAddress:Craig_Wayman@Typex.com");
         eid.addAlias("Craig Michael Wayman");
         eid.addAlias("Craig M Wayman");
      eid = d.addEid(eidname2,eiddesc2);
         eid.addAssociation(Association.EIM_TARGET,"My_iSeries","BrownC");
         eid.addAssociation(Association.EIM_TARGET,"LDAP","charlie");
         eid.addAssociation(Association.EIM_SOURCE,"My_AIX","nworBC");
         eid.addAssociation(Association.EIM_TARGET,"My_RACF","eilrahc");
         eid.addAdditionalInfo("emailAddress:CharlieBrown@Snoopy.com");
         eid.addAlias("Charlie \"Legend\" Brown");
      eid = d.addEid(eidname3,eiddesc3);
         eid.addAssociation(Association.EIM_SOURCE,"My_iSeries","Eric");
         eid.addAssociation(Association.EIM_TARGET,"LDAP","eRiC");
         eid.addAssociation(Association.EIM_SOURCE,"My_AIX","cirE");
         eid.addAssociation(Association.EIM_ADMIN,"My_RACF","RiCe");
         eid.addAdditionalInfo("emailAddress:Eric@29AcaciaRoad.com");
         eid.addAlias("Bananaman");
      eid = d.addEid(eidname4,eiddesc4);
         eid.addAssociation(Association.EIM_ADMIN,"My_iSeries","KentC");
         eid.addAssociation(Association.EIM_TARGET,"LDAP","clark");
         eid.addAssociation(Association.EIM_SOURCE,"My_AIX","Ctnek");
         eid.addAssociation(Association.EIM_TARGET,"My_RACF","kralc");
         eid.addAdditionalInfo("emailAddress:CK@DailyPlanet.com");
         eid.addAdditionalInfo("emailAddress:Superman@SaveThePlanet.com");
         eid.addAlias("Superman");
      return d.getEids();
   }
   catch (EimException e)
   {
      System.out.println("Unable to add eid.");
      e.printStackTrace();
      }
      return null;
   }
```

9.4.11 The outputDomainInfo method

This is the start of the methods which show you how to retrieve information stored about the different objects in EIM. I have designed these output methods so that they are recursive. If the boolean parameter is set then this will cause the output to display information about registries and from that registry output method it will recurse to show the different registry users associated with that registry. For each of the methods listed in these output methods there is either a set, get or add method associated with them. Some information cannot be set and can only be retrieved. An example of this in the outputDomainInfo method is the getHost method which returns the name of the system where the EIM domain resides.

Example 9-11 The outputDomainInfo method

```
   private void outputDomainInfo(Domain d, boolean recursive)
   {
      try
      {
         System.out.println("Domain Information for :" + d.getName());
         System.out.println("\tDescription: " + (d.getDescription()=="" ? "*No Description 
Available*" : d.getDescription()));
         System.out.println("\tDistinguished Name: " + d.getDn());
         System.out.println("\tHost: " + d.getHost());
         System.out.println("\tName: " + d.getName());
         System.out.println("\tPort: " + d.getPort());
         System.out.println("\tURL: " + d.getUrl());
         System.out.println("\tisConected: " + d.isConnected());
         System.out.println("\tisSSL: " + d.isSSL());
         System.out.println("");

         if(recursive)
         {
            Set registries = getRegistries(d, false);
            outputRegistryInfo(registries, recursive, true);
         }
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving domain information");
         e.printStackTrace();
      }
   }
```

For each of these output methods I have also other methods with the same name but instead of taking in only a single object these methods take it a java.util.Set object. This method then iterates over this set calling the output methods similar to the method above which outputs the information for a given object. These additional methods allow for a much greater capacity for reporting the information available to us about the domains. Here is an example of the corresponding getDomaininfo method which takes in a Set object of domain names.

Example 9-12 The outputDomainInfo method with Set capabilities

```
   private void outputDomainInfo(Set domains, boolean recursive, boolean showEIDs, boolean 
eidRecurse)
   {
      try
      {
         Iterator it = domains.iterator();
         while(it.hasNext())
         {
            Domain d = getDomain((String)it.next(),false);
            outputDomainInfo(d,recursive);
            if(showEIDs)
            {
               Set eids = getEids(d, false);
               outputEidInfo(eids, eidRecurse);
            }
         }
      }
      catch(EimException e)
      {
         e.printStackTrace();
      }
   }
```

This method also takes in another parameter called showEids. In the EIM implementation the identifiers are separate entities to the registries and as such are not caught by the recursive nature of the methods I have created, to combat this I have created this boolean flag which will if desired output a list of identifiers. Depending on the eid recurs boolean it may also output the associations for that identifier.

Output

This is an example of the domain information that is outputted by the outputDomainInfo method:

Example 9-13 outputDomainInfo output

```
Domain Information for :BlueNotesEIMDomain
   Description: Created By Blue Notes EIM Administrator (BlueNotes.com)
   Distinguished Name: ibm-eimdomainname=BlueNotesEIMDomain
   Host: as20.itsoroch.ibm.com
   Name: BlueNotesEIMDomain
   Port: 389
   URL: ldap://as20.itsoroch.ibm.com:389/ibm-eimdomainname=BlueNotesEIMDomain
   isConected: true
   isSSL: false
```

9.4.12 The outputRegistryInformation method

This method takes in an individual registry object and displays all the available information about it. The method first displays all the information available from the registry class. It then gets any aliases that the registry may have. There are two types of registry in EIM, application and system. These different interfaces also have different methods, the code decides which type of registry the object is and then casts it to its associated interface. The methods for this object can then be called.

In this method you can see the benefit of the hashmaps to translate the information into a usable format. Another feature of this reporting structure is that if the method is called from a previous output method the `fromDomainOutput` boolean variable is set to true. This determines the number of tabs that the output is indented by, this allows for a nice hierarchical structure to the output which makes it easier to read, decipher and find where objects are located in the EIM tree.

Example 9-14 The outputregistryInfo method

```
private void outputRegistryInfo(Registry reg, boolean fromDomainOutput, boolean recursive)
{
   try
   {
      String tab;

      if(fromDomainOutput)
         tab = "\t\t";
      else
         tab = "\t";

      System.out.println((fromDomainOutput ? "\t" : "") + "Registry Information for: " + reg.getName());
      System.out.println(tab + "Description: " + (reg.getDescription()=="" ? "*No Description Available*"
: reg.getDescription()));
      System.out.println(tab + "Name: " + reg.getName());
      System.out.println(tab + "Type: " + reg.getType());
      System.out.println(tab + "Type Translated: " + rh.get(reg.getType()));
      System.out.println(tab + "Kind: " + reg.getKind());
      System.out.println(tab + "Kind Translated: " + rh.get(new Integer(reg.getKind())));
      System.out.println(tab + "Uuid: " + reg.getUuid());
      Set aliases = reg.getAliases();
      if(!aliases.isEmpty())
         outputRegistryAliasInformation(aliases, (fromDomainOutput) ? "\t\t" : "\t");

      switch(reg.getKind())
      {
         case Registry.EIM_SYSTEM_REGISTRY: //System Registry
            SystemRegistry sysReg = (SystemRegistry)reg;
            System.out.println(tab + "Uri: " + (sysReg.getUri()=="" ? "*No Uri Defined*" :
sysReg.getUri()));
            break;

         case Registry.EIM_APPLICATION_REGISTRY: //Application Registry
            ApplicationRegistry appReg = (ApplicationRegistry)reg;
            System.out.println(tab + "SystemRegistry Name: " + appReg.getSystemRegistryName());
            break;
      }

      System.out.println("");

      if(recursive)
         outputRegistryUsersInfo(reg, recursive);
   }
   catch (EimException e)
   {
      System.out.println("Error retrieving registry information");
      e.printStackTrace();
   }
}
```

Here is the method which is called when we are using a java.util.Set of registry objects:

Example 9-15 The outputRegistryInfo with Set capabilities

```
private void outputRegistryInfo(Set reg, boolean fromDomainOutput, boolean recursive)
{
   Iterator it = reg.iterator();
   while(it.hasNext())
      outputRegistryInfo((Registry)it.next(),fromDomainOutput,recursive);
   }
```

Output

Here is an example of the registry information outputted by the outputRegistryInfo method. This example outputs an application registry.

Example 9-16 outputRegistryInformation output

```
   Registry Information for: LDAP
      Description: My LDAP Server
      Name: LDAP
      Type: 1.3.18.0.2.33.7-caseIgnore
      Type Translated: LDAP
      Kind: 2
      Kind Translated: Application Registry
      Uuid: ef832800-777e-186e-8a4e-0004ac018327
      Alias information:
         Value: host.network.com Type: DNSHostName
         Value: 91.92.94.95   Type: UserDefinedInAPIs
      SystemRegistry Name: My_iSeries
```

9.4.13 The outputRegistryAliasInformation method

Unlike the aliases applied to EIM identifiers, registry aliases are formed using a RegistryAlias object. When we get a list of a registries aliases from the getAliases method seen above a java.util.Set is returned containing RegistryAlias objects. Because the getAliases method returns a set we have no choice but to create an output method which uses a java.util.Set object as a parameter, or some code which iterates over this java.util.Set object, a method makes perfect sense though as it makes the code reusable. The tabs parameter is passed to that the information passed can be presented in a hierarchy.

Example 9-17 The outputRegistryAliasInformation method with Set capabilities

```
   private void outputRegistryAliasInformation(Set regAliases, String tabs)
   {
      Iterator it = regAliases.iterator();
      System.out.println(tabs + "Alias information:");
      while(it.hasNext())
         outputRegistryAliasInformation((RegistryAlias)it.next(),tabs);
   }
```

In keeping with the previous methods where we iterate over objects, the output method is kept separate from the outputted information method we created another method which takes this RegistryAlias object and outputs its information.

Example 9-18 The outputRegistryAliasInformation method

```
   private void outputRegistryAliasInformation(RegistryAlias regAlias, String tabs)
   {
      System.out.println(tabs + "\t" + "Value: " + regAlias.getName() + "   Type: " +
regAlias.getType());
   }
```

The output for these methods can be seen near the bottom of the registry output example above.

Important: Remember that in this release of EIM only the six pre-defined registry alias types can be seen in iSeries navigator. Any attempt to use a different value either through code or iSeries navigator results in the alias type to be set to TCP/IP. Getting the registry alias information through the method above returns the user defined alias type. To fix this problem it is recommended to update the version of iSeries Access to be service level SI09809.

9.4.14 The outputRegistryUserInfo method

You may have seen in the above outputRegistryInfo method that there is a method call at the bottom of the method based on the recursive boolean parameter. This method call accesses the outputRegistryUserInfo method with a java.util.Set object containing a list of RegistryUser objects. Like the registry alias method we use a method that takes in a java.util.Set object and iterates over it calling a method which will output the information stored in the RegistryUser object. When listing the registry users for a given registry only the target associations are returned. This is because a user cannot access a system using a source association or an administrative association.

Example 9-19 outputRegistryUsersInfo method with Set capabilities

```
private void outputRegistryUsersInfo(Registry reg, boolean fromRegistryOutput)
{
   try
   {
      Iterator it = reg.getUsers().iterator();
      while(it.hasNext())
         outputRegistryUserInfo((RegistryUser)it.next(),fromRegistryOutput);
   }
   catch(EimException e)
   {
      System.out.println("Error accessing registry user");
      e.printStackTrace();
   }
}
```

This is the method which is called to output the information in the RegistryUser object:

Example 9-20 The outputRegistryUserInfo method

```
   private void outputRegistryUserInfo(RegistryUser regUsr, boolean fromRegistryOutput)
   {
      try
      {
         String tab;

         if(fromRegistryOutput)
            tab = "\t\t\t";
         else
            tab = "\t\t";

         System.out.println((fromRegistryOutput ? "\t\t" : "\t") + "RegistryUser Information for this 
registry");
         System.out.println(tab + "Description: " + (regUsr.getDescription()=="" ? "*No Description 
Available*" : regUsr.getDescription()));
         System.out.println(tab + "Registry Name: " + regUsr.getRegistryName());
         System.out.println(tab + "Target User Name: " + regUsr.getTargetUserName());
         System.out.println("");
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving registry information");
         e.printStackTrace();
      }
   }
```

Output

This is the information that is outputted from the outputRegistryUserInfo method:

Example 9-21 The outputRegistryUserInfo method output

```
RegistryUser Information for this registry
   Description: *No Description Available*
   Registry Name: LDAP
   Target User Name: CRAIG
```

9.4.15 The outputEidInfo method

When a list of Eids is retrieved from the domain getEids method a java.util.Set object is returned containing a list of Eid objects. In keeping with the previous design of writing a method which iterates over this java.util.Set object and another which outputs the information we have the following two methods available to us:

Example 9-22 The outputEidInfo method with Set capabilities

```
private void outputEidInfo(Set eids, boolean recurse)
{
   Iterator it = eids.iterator();
   while(it.hasNext())
      outputEidInfo((Eid)it.next(), recurse);
}
```

In addition to the identifier object to output this method also takes a boolean parameter which determines whether the associations for this identifier should be output. Here is the method which output the Eid information:

Example 9-23 The outputEidInfo method

```
private void outputEidInfo(Eid eid, boolean recurse)
{
   try
   {
      System.out.println("Eid information for: " + eid.getName());
      System.out.println("\tName: " + eid.getName());
      System.out.println("\tDescription: " + eid.getDescription());
      System.out.println("\tUuid: " + eid.getUuid());
      Set aliases = eid.getAliases();
      if(!aliases.isEmpty())
         outputStringInformation(aliases, "Alias", "\t");
      Set additionalInfo = eid.getAdditionalInfo();
      if(!additionalInfo.isEmpty())
         outputStringInformation(additionalInfo, "Additional", "\t");
      System.out.println("");

      if(recurse)
         outputAssociationInfo(eid.getAssociations(Association.EIM_ALL_ASSOC), true);
   }
```

```
      catch (EimException e)
      {
         System.out.println("Error retrieving eid information");
         e.printStackTrace();
      }
   }
```

Output

Here is an example of the output from an identifier object:

Example 9-24 The outputEidInfo output

```
Eid information for: Clark Kent1068179425563
   Name: Clark Kent1068179425563
   Description: Better than Eric?
   Uuid: a6d41800-7aac-186e-8a4e-0004ac018327
   Alias information:
      Alias: Clark Kent1068179425563
      Alias: Superman
      Alias: Clark Kent
   Additional information:
      Additional: emailAddress:CK@DailyPlanet.com
      Additional: emailAddress:Superman@SaveThePlanet.com
```

In this output you can also see the aliases and the additional information for this user. We explain how this is output in the following example.

9.4.16 The outputStringInformation method

In the examples above we have needed to write a method for each java.util.Set of objects to be outputted. In the identifier object there are two methods which can be seen in Example 9-23, these are the getAdditionalInfo and getAliases methods. Both these methods return a java.util.Set object containing a list of strings. Instead of writing two separate methods to output this information we have created a generic output string information method. This takes in the list of object, the label to use for output and also a tabs parameter to allow for indentation in the hierarchy format. Here is the generic string output method:

Example 9-25 The outputStringInformation method

```
   private void outputStringInformation(Set info, String label, String tabs)
   {
      Iterator it = info.iterator();
      System.out.println(tabs + label + " information:");
      while(it.hasNext())
         System.out.println(tabs + "\t" + label + ": " + ((String)it.next()));
   }
```

The output for this method can be seen in the identifier output example above under the alias information and additional information headings.

9.4.17 The outputAssociationInfo method

This is the last of the output methods. This method as its name suggests outputs the association information for a given identifier. The getAssociations method of the identifier object is used to get a java.util.Set object containing a list of Association objects. Here are the code examples which iterate over this list of associations, the method which outputs the information contained within this association object and an example of the output which is shown to the user:

Example 9-26 The outputAssociationInfo method with Set capabilities

```
   private void outputAssociationInfo(Set assoc, boolean fromEidOutput)
   {
      Iterator it = assoc.iterator();
      while(it.hasNext())
         outputAssociationInfo((Association)it.next(), fromEidOutput);
   }
```

Example 9-27 The outputAssociationInfo method

```
   private void outputAssociationInfo(Association assoc, boolean fromEidOutput)
   {
      try
      {
         String tab;

         if(fromEidOutput)
            tab = "\t\t";
         else
            tab = "\t";
         System.out.println((fromEidOutput ? "\t" : "") + "Association information for: "
+ (assoc.getEid()).getName());
         System.out.println(tab + "Association Type: " + assoc.getAssociationType());
         System.out.println(tab + "Association Type Translation: " + h.get(new
Integer(assoc.getAssociationType())));
         System.out.println(tab + "Registry: " + assoc.getRegistryName());
         System.out.println(tab + "User ID: " + assoc.getUid());
         System.out.println("");
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving eid information");
         e.printStackTrace();
      }
   }
```

Output

This is an example of the association information which is outputted from a call in the outputEidInfo method. To make this information more meaningful I have shown the association information relating to a particular identifier object. In the registry user output we see that only target associations were returned, this is different when we list the associations relating to a particular user. When we list the associations relating to a particular identifier we are doing just that, listing associations, we are not using the returned values to gain access to a server or service. In the example below we see all associations that this identifier has in this EIM domain.

Example 9-28 The outputAssociationInfo method output

```
Eid information for: Eric
   Name: Eric
   Description: When Eric eats a banana..
   Uuid: 6ac53800-777f-186e-8a4e-0004ac018327
   Alias information:
      Alias: Bananaman
      Alias: Eric
```

```
Additional information:
   Additional: emailAddress:Eric@29AcaciaRoad.com

Association information for: Eric
   Association Type: 1
   Association Type Translation: target
   Registry: LDAP
   User ID: ERIC

Association information for: Eric
   Association Type: 2
   Association Type Translation: source
   Registry: My_AIX
   User ID: cirE

Association information for: Eric
   Association Type: 2
   Association Type Translation: source
   Registry: My_iSeries
   User ID: ERIC

Association information for: Eric
   Association Type: 4
   Association Type Translation: admin
   Registry: My_RACF
   User ID: RICE
```

9.4.18 The deleteEIMDomain method

When writing our code we found that it was useful to have a delete method. This delete method was created primarily to cleanse information from the EIM domain so that as the code example evolved we could add new information to the objects held in EIM and quickly add new objects to the domain without writing extra code, the EIM domain would be cleansed and then re-created. This deleteEIMDomain method proved very useful and also generated some interesting results which is why it is included in this section.

By writing this code example we found that in order to delete an EIM domain all objects in the EIM domain needed to be removed first. In the case of registries there is also a specific order to this deletion because of the way registries are associated with each other. When an application registry is added to EIM it references a system registry. If a system registry is attempted to be deleted then it will cause an exception if there are any application registries associated with it. For this reason we first remove application registries and then system registries.

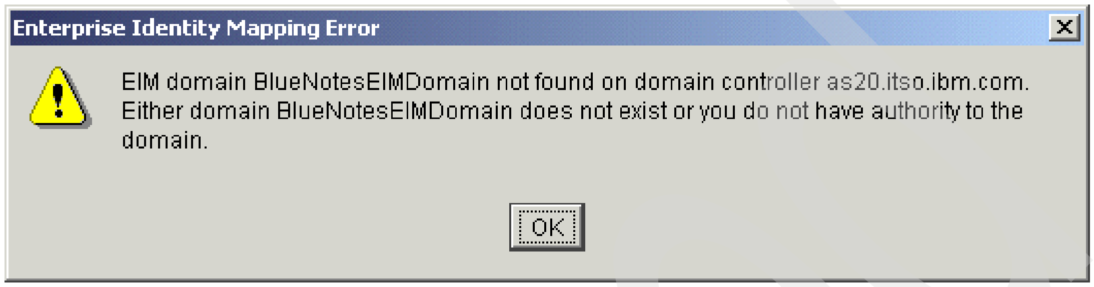
In addition to these two types of registries there is also another property which allows us to retrieve both types of registry. This property is used in the domain getRegistry methods to determine which registries to return. For example if an application registry and a system registry had the same name then this flag could be use to return both registries or either of the two options. For completeness we then retrieve all remaining registries, this should always return an empty java.util.Set object but allows for extensibility to delete registries if different types of registries are changed in the future. Assuming that if a new registry type is added it does not relate to an application registry or a system registry.

The delete code example shown below takes in a domain name string as a parameter. This is then used to retrieve the domain and start the deleting of information contained within it.

Example 9-29 The deleteEIMDomain method

```
private void deleteEimDomain(String domainName)
{
   try
   {
      d = getDomain(domainName,false);
      if(d!=null)
      {
         Set eids = getEids(d, false);
         Iterator it = eids.iterator();
         while(it.hasNext())
         {
            Eid eid = ((Eid)it.next());
            eid.delete();
         }

         //get registries
         Registry temp;
         Set registries = getRegistries(d, false);
         it = registries.iterator();
         //Must first remove appl registries as reference system registries
         while(it.hasNext())
         {
            temp = (Registry)it.next();
            if(temp.getKind()==Registry.EIM_APPLICATION_REGISTRY)
               temp.delete();
         }

         //Then remove the systems
         registries = getRegistries(d, false);
         it = registries.iterator();
         while(it.hasNext())
         {
            temp = (Registry)it.next();
            if(temp.getKind()==Registry.EIM_SYSTEM_REGISTRY)
               temp.delete();
         }

         //In case there are any not defined as either type
         registries = getRegistries(d, false);
         it = registries.iterator();
         while(it.hasNext())
         {
            temp = (Registry)it.next();
            if(temp.getKind()==Registry.EIM_ALL_REGISTRIES)
               temp.delete();
         }

         d.delete();
         System.out.println("Delete Successfull");
      }
      else
         System.out.println("Canont retrieve domain object");
   }
   catch(EimException e)
   {
      System.out.println("Cannot complete delete successfully");
      e.printStackTrace();
   }
}
```

If before running the deleteEIMDomain method the EIM domain was added and browsable from the iSeries Navigator then an error message will occur if the domain is selected again. This is because the EIM domain has been deleted. We are showing you this error so that if it occurs you know that this message could mean that the domain has been deleted. This error can be seen in Figure 9-1.


Enterprise Identity Mapping Error

EIM domain BlueNotesEIMDomain not found on domain controller as20.itso.ibm.com. Either domain BlueNotesEIMDomain does not exist or you do not have authority to the domain.

Figure 9-1 iSeries Navigator Error

9.4.19 The startReport method

This is one of the last methods in the ReportEIM class. When the ReportEIM class is instantiated this is the method which is called to produce the report. Based on the report methods which are contained within this class a variety of reports can be created. They are by no means limited to the reports which we are documenting in this section. The startReport method takes an integer parameter. The options are defined as public constants in the class:

- ReportOne
- ReportTwo
- ReportThree
- ReportFour
- DeleteDomain

A switch statement is performed on this parameter to determine which report to create. Here is the code for this method:

Example 9-30 The startReport method

```
public void startReport(int reportToRun)
{
   //Setup Hash Tables For Quick Lookup Of Constants
   createAssociationTypeHashMap();
   createRegistryTypeHashMap();

   switch(reportToRun)
   {
      case ReportOne:
         reportOne();
      break;

      case ReportTwo:
         reportTwo();
      break;

      case ReportThree:
         reportThree();
      break;
```

```
        case ReportFour:
           reportFour();
        break;

        case DeleteDomain:
           //Method used to delete all elements in a domain
           deleteEimDomain("ibm-eimdomainname=BlueNotesEIMDomain");
        break;
     }
  }
```

This is a good way of creating different reports and segregates the information contained within the reports so new reports can be added relatively simply remembering also to create a public constant so that this new method can be called and its corresponding entry created in the switch statement.

Finally here is the code for the different reports that can be generated. The explanation for these reports are listed as comments in the methods themselves.

Example 9-31 The four reports which can be created

```
   private void reportOne()
   {
      try
      {
      /* This report first creates the information in EIM based on the default information
         Contained within this class and then creates a call which will display all
information
         about the domain , its registries and the registry users for those registries*/
         Domain d = getDomain("ibm-eimdomainname=BlueNotesEIMDomain",true);
         Set registries = createRegistries(d);
         Set eids = createEids(d);
         outputDomainInfo(d, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }

   private void reportTwo()
   {
      try
      {
      /* This report retrieves an existing domain object and displays information about
the domain only
         A list of registries is then retrived and this information is outputted*/
         Domain d = getDomain("ibm-eimdomainname=BlueNotesEIMDomain",false);
         outputDomainInfo(d, false);
         Set registries = getRegistries(d, false);
         outputRegistryInfo(registries, false, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }
   private void reportThree()
   {
      try
      {
      /* This report accesses a domain object and outputs all the identifiers and
         their associations contained within the EIM domain*/
         Domain d = getDomain("ibm-eimdomainname=BlueNotesEIMDomain",false);
         Set eids = getEids(d, false);
         outputEidInfo(eids, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }

   private void reportFour()
   {
      try
      {
      /* This report retrieves a list of all the EIM domains contained in the LDAP server
         and outputs all information about the domains, registries, registry users and
         identifiers with their associations                    */
         Set domains = getAllDomains();
         outputDomainInfo(domains, true, true, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }
}
```

Remember a full source listing of this code and the full output from all four reports can be found in Appendix F, “Java code listings and output examples” on page 269.

9.5 Java example: EIMAuthorities

This class returns information about the different authorities in EIM. A list of different user authorities can be seen in the Enterprise Identity Mapping chapter. The different levels of security can be related to specific methods and classes as seen in 9.3, “Security in the Java classes” on page 192. Compared with the ReportEIM class seen on the previous pages this class is simple in contrast. There are only two classes in EIM which allow use to access security information about certain authorities. These are the AccessContext and UserAccess classes. A brief description of these classes can be found in 9.2, “Java classes and interfaces” on page 188.

I have written this class to show the available authorities in the EIM domain. At this release the iSeries Navigator only allows you to configure EIM authority information it does not provide functionality to list the different authorities or information about them. This is a serious consideration for auditing access privileges and accountability for actions. Hopefully this class will go some way to providing this information to you.

9.5.1 The createEIMAuthoritiesHashMap method

As in the previous ReportEIM class I have created a hash map object to allow the quick retrieval and translation of EIM authority constant values into a readable format. This code example is listed below.

Example 9-32 The createEIMAuthoritiesHashMap

```
   public void createEIMAuthoritiesHashmap()
   {
      ah = new HashMap();
      ah.put(new Integer(AccessContext.EIM_ACCESS_ADMIN),"Administrative Access");
      ah.put(new Integer(AccessContext.EIM_ACCESS_DN),"User Type For LDAP In Distinguished Name Format");
      ah.put(new Integer(AccessContext.EIM_ACCESS_IDENTIFIER_ADMIN),"Identifier Access");
      ah.put(new Integer(AccessContext.EIM_ACCESS_KERBEROS),"User Type For LDAP In Kerberos Principal 
Format");
      ah.put(new Integer(AccessContext.EIM_ACCESS_MAPPING_LOOKUP),"Mapping Lookup Access");
      ah.put(new Integer(AccessContext.EIM_ACCESS_REG_ADMIN),"All Registry Access");
   }
```

9.5.2 Using the AccessContext class

After a domain object has been retrieved using the DomainManager class the getAccessContext method can then be called to retrieve an AccessContext object.

Example 9-33 Accessing an AccessContext object

```
   AccessContext ac = d.getAccessContext();
```

This AccessContext class is a very powerful class despite its small number of methods and properties and can be used to access information about all authorities which use EIM. It is possible for a complete EIM authorities administration tool to be built around this class.

There is one method in this class which allows us to access lists of the different EIM authorities in a domain. This is the getAdminAccessUser method and it uses the AccessContext properties listed in the hash map method above to determine which authorities information to return. It is worth making a note at this point that two properties listed above, AccessContext.EIM_ACCESS_DN and AccessContext.EIM_ACCESS_KERBEROS, are not EIM authorities but they are used to retrieve UserAccess objects which we see in the Using the UserAccess class section below.

This code example is actually part of another method. The complete code listing for this EIMAuthorities class can be seen in Appendix F, “Java code listings and output examples” on page 269.

Example 9-34 getAdminAccessUsers examples

```
         System.out.println("EIM Authority Information:");
         System.out.println("");

         Set temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_ADMIN);
         outputStringInformation(temp, "Admin User", "\t");

         temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_IDENTIFIER_ADMIN);
         outputStringInformation(temp, "Identifier Admin User", "\t");

         temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_MAPPING_LOOKUP);
         outputStringInformation(temp, "Mapping Lookup User", "\t");
```

```
        temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_REG_ADMIN);
        outputStringInformation(temp, "Registry Admin User", "\t");
```

You can see that this code example uses a method from the ReportEIM class, the outputStringInformation method. This proves the benefit of writing extensible methods to perform common functions within a class.

Output

The output for this code example is very interesting. First we display the output and then discuss it.

Example 9-35 getAdminAccessUsers example output

```
EIM Authority Information:

   Admin User information:
      Admin User: os400-profile=itscid12,cn=accounts,os400-sys=as20.itso.ibm.com
      Admin User: cn=administrator
      Admin User: ibm-eimdomainname=bluenoteseimdomain

   Admin Identifier User information:
      Identifier Admin User: ibm-kn=krbsvr400
      Identifier Admin User: ibm-eimdomainname=bluenoteseimdomain

   Mapping Lookup User information:
      Mapping Lookup User: os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.com
      Mapping Lookup User: ibm-kn=krbsvr400
      Mapping Lookup User: ibm-eimdomainname=bluenoteseimdomain

   Registry Admin User information:
      Registry Admin User: os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.com
      Registry Admin User: ibm-eimdomainname=bluenoteseimdomain
```

To explain this output, we must first look at the information which was added into the EIM authorities. This information was added through the iSeries navigator:

Table 9-7 Authority information for BlueNotesEIMDomain

User	Type	Authority Access
cn=administrator	distinguished name	EIM_ACCESS_ADMIN
krbsvr400	kerberos principal	EIM_ACCESS_IDENTIFIER_ADMIN EIM_ACCESS_MAPPING_LOOKUP
itscid12	OS400 user profile	EIM_ACCESS_ADMIN
itscid11	OS400 user profile	EIM_ACCESS_MAPPING_LOOKUP EIM_ACCESS_REG_ADMIN
itscid10	OS400 user profile	Does not have a defined access level but is a registry administartor for specific registries: ► LDAP ► My_iSeries ► My_AIX

When a EIM_ACCESS_ADMIN user is created the user has permissions to perform the functions of all the other authorities. This however is not shown in the output for the other authorities. The information which is returned only relates to that specific authority rather than all users who have permissions to perform that authority. It is also interesting to see in the output that the EIM domain has access to perform all functions it desires without having been added to any specific authority.

We now move on to retrieving a list of registry administrators for specific registries. This code example shows us retrieving a list of registry administrators for the My_iSeries registry and the My_RACF registry.

Example 9-36 Accessing registry associated authorities

```
      System.out.println("EIM Registry Authority Information:");
      System.out.println("");

      temp = ac.getRegistryAccessUsers("My_iSeries");
      outputStringInformation(temp, "My_iSeries Registry Admin User", "\t");

      temp = ac.getRegistryAccessUsers("My_RACF");
      outputStringInformation(temp, "My_RACF Registry Admin User", "\t");
```

The output can be seen below. In the authority table above we defined user itscid10 as an administrator for specific registries, one of which was the My_iSeries registry. We did not however define him as a registry administrator for the My_RACF registry hence the user is not listed there. It is also worth noting that although the user itscid11 was given the registry administrator authority for all registries the user is not listed here for either registry. This is because of a security concern, if a user has access to all registries in the EIM domain does this user need to be listed for each registry. The users who have registry administration privileges can be accessed through the getAdminAccessUsers method with the relevant parameter. The EIM domain also has access to all the registries listed here. You may be thinking that the EIM domain was listed as a registry administrator so why is it listed here if only specific registry administrators are listed. The answer is that the EIM domain can access all objects contained in the EIM domain.

Example 9-37 Registry authority for specific registries

```
EIM Registry Authority Information:

   My_iSeries Registry Admin User information:
      My_iSeries Registry Admin User: os400-profile=itscid10,cn=accounts,os400-sys=as20.itso.ibm.com
      My_iSeries Registry Admin User: ibm-eimdomainname=bluenoteseimdomain

   My_RACF Registry Admin User information:
      My_RACF Registry Admin User: ibm-eimdomainname=bluenoteseimdomain
```

9.5.3 Using the UserAccess class

This method follows the output method outline as detailed in the ReportEIM class. Normally when we had a method like this we wrote another method to allow for a situation when multiple objects are returned in a java.util.Set object. The methods in the AccessContext class which return a UserAccess object only ever returns a single UserAccess object. This makes another method to output multiple UserAccess objects redundant. This is the code example which outputs the information about the UserAccess object.

Example 9-38 The outputUserAccessInformation method

```
   private void outputUserAccessInformation(UserAccess ua, String name, String tabs)
   {
      System.out.println("Output User Access Information for " + name);
      System.out.println(tabs + "\t" + "Dn: " + ua.getDn());
      Set temp = ua.getRegistries();
      if(!temp.isEmpty())
         outputStringInformation(temp, "Admin authority to following Registries", tabs +
"\t");
      System.out.println(tabs + "\t" + "hasAdmin: " + ua.hasAdmin());
      System.out.println(tabs + "\t" + "hasEidAdmin: " + ua.hasEidAdmin());
      System.out.println(tabs + "\t" + "hasRegistryAdmin: " + ua.hasRegistryAdmin());
      System.out.println(tabs + "\t" + "hasMappingLookup: " + ua.hasMappingLookup());
      System.out.println("");
   }
```

By outputting this information some very useful information can be seen about the way authorities can be used and are stored in EIM.

This is the code which is used to retrieve UserAccess object from the AccessContext class:

Example 9-39 AccessUser information

```
         System.out.println("User Context Information");
         System.out.println("");

         UserAccess ua =
ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"os400-profile=itscid12,cn=accounts,os400-sys=as20.itso.ibm.co
m");
         outputUserAccessInformation(ua, "itscid12", "\t");

         ua =
ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.co
m");
         outputUserAccessInformation(ua, "itscid11", "\t");

         ua =
ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"os400-profile=itscid10,cn=accounts,os400-sys=as20.itso.ibm.co
m");
         outputUserAccessInformation(ua, "itscid10", "\t");

         ua = ac.getUserAccess(AccessContext.EIM_ACCESS_KERBEROS,"krbsvr400");
         outputUserAccessInformation(ua, "krbsvr400", "\t");

         ua = ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"cn=administrator");
         outputUserAccessInformation(ua, "cn=administrator", "\t");
```

When retrieving a users AccessUser object the type of the user name and the name of the user are specified. The type of the user name determines the format of what should be specified to retrieve that users information. For example above in a distinguished format or in a kerberos principal format. In the iSeries navigator there is a third option to use an existing OS400 user profile. These different types for names all resolve to a distinguished names, however it is encapsulated using the user interface in iSeries Navigator to simplify the process of typing in distinguished names for all users with authorities in the database.

For example it is possible to retrieve the kerberos principal using a distinguished name format. Either of these methods give the same output for this users authority. The distinguished name for this kerberos principal was retrieved from the output of the first kerberos code example shown here and tested using the EIM_ACCESS_DN property, this works without error although it could be a best practice and also simpler to refer to kerberos principals using the first method.

Example 9-40 Kerberos principal and distinguished name

```
        ua = ac.getUserAccess(AccessContext.EIM_ACCESS_KERBEROS,"krbsvr400");
        outputUserAccessInformation(ua, "krbsvr400", "\t");

        ua = ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"ibm-kn=krbsvr400");
        outputUserAccessInformation(ua, "krbsvr400", "\t");
```

Output

This output is split into five parts. Each represents a user from the EIM authorities table previously seen in this chapter.

The first user explained is itscid12 who as the EIM_ACCESS_ADMIN authority. The Dn options shows this account as being an AS400 user profile. The subsequent details outputted for this user are booleans based on methods with the same name as the label they represent. As explained in the getAdminAccessUsers chapter even though the admin authority is able to perform all authorities actions the information is authority specific only returning information pertaining to that level of access not an overall access level in the system.

The second is for the OS400 user profile itscid11. This user is a registry administrator for all registries and can perform mapping lookup operations. In the third example we see that user itscid10 has registry administrator access to the three registries listed. Even though user itscid11 has the registry administrator authority for all registries no registries are listed here.

The fourth example shows the output for a kerberos principal name.

The fifth example shows us outputting UserAccess information based on a supplied distinguished name. I have included this example for completeness so that the output shows:

- An OS400 user profile distinguished name
- A Kerberos principal
- A distinguished name

You can see from the Dn attribute of each output that no matter how the information is added either through code or from the iSeries Navigator all entries resolve to a distinguished name.

Example 9-41 Output from outputUserAccessInformation method

```
Output User Access Information for itscid12
      Dn: os400-profile=itscid12,cn=accounts,os400-sys=as20.itso.ibm.com
      hasAdmin: true
      hasEidAdmin: false
      hasRegistryAdmin: false
      hasMappingLookup: false

Output User Access Information for itscid11
      Dn: os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.com
      hasAdmin: false
      hasEidAdmin: false
      hasRegistryAdmin: true
      hasMappingLookup: true

Output User Access Information for itscid10
      Dn: os400-profile=itscid10,cn=accounts,os400-sys=as20.itso.ibm.com
      Admin authority to following Registries information:
         Admin authority to following Registries: LDAP
         Admin authority to following Registries: My_iSeries
         Admin authority to following Registries: My_AIX
      hasAdmin: false
      hasEidAdmin: false
      hasRegistryAdmin: false
      hasMappingLookup: false

Output User Access Information for krbsvr400
      Dn: ibm-kn=krbsvr400
      hasAdmin: false
      hasEidAdmin: true
      hasRegistryAdmin: false
      hasMappingLookup: true

Output User Access Information for cn=administrator
      Dn: cn=administrator
      hasAdmin: true
      hasEidAdmin: false
      hasRegistryAdmin: false
      hasMappingLookup: false
```

Remember that the entire code listing and sample output are available in Appendix F, "Java code listings and output examples" on page 269.

9.6 Kerberizing an application

Unfortunately we are not going to include an example of enabling an application for authentication using Kerberos in this chapter however there are good sites on the Internet which are able to show you how. On the Sun Web site there are tutorials for kerberizing applications with comprehensive code examples and help available. These tutorials use the Java Generic Security Service (GSS) APIs and the Java Authentication and Authorization Service (JAAS) APIs in the code examples. There are tutorials for using both APIs or either providing an excellent knowledgebase including everything you will need to know to enable authentication using kerberos and in the future other authentication mechanisms. These tutorials detail examples using logon screens and pure code to authenticate users providing a number of ways to factor authentication functions into your applications. These tutorials can be found at the following link.

http://java.sun.com/j2se/1.4.2/docs/guide/security/jgss/tutorials/

9.7 C EIM API

In this we give examples of how to use the C APIs. We are not going to specify information about the C APIs themselves. This information already exists on the iSeries Information Center on the IBM Web site and is extremely comprehensive. This C information deals with code examples, information about each method and also the return types available from each method. This information about the C APIs is the equivalent of the Javadocs for the eim jar and in some cases is a lot more comprehensive.

To access the iSeries Information Center, follow the directions here:

- Open a Web browser and enter the URL:

  http://www.ibm.com/eServer/iSeries/infocenter

- This will take you to the iSeries Information Center homepage. From here you select the region you are living in. This takes you to a table of languages where you can select the language in which you can view the pages.

- From the links page on the left of the window, select: **Network -> Networking Security -> Enterprise Identity Mapping (EIM) -> APIs for EIM**.
- This brings up a small page detailing information about the APIs. At the bottom of this page there is a link in the last sentence ‘see the Enterprise Identity Mapping (EIM) topic’. When this link is clicked it takes you to a page where information about the C APIs can be retrieved. Alternatively from the main Information Center page, use the following steps. Select **Programming -> APIs By Category**, from here click the **security** link, and then click the **Enterprise Identity Mapping (EIM) APIs** link.

An alternative way to access this C EIM API information is to click the following link:

http://publib.boulder.ibm.com/iseries/v5r2/ic2924/info/apis/sec5.htm

9.8 C Generic Security Service (GSS) API

The GSS APIs are an implementation of various RFCs which provide authentication functions for client server interactions. Coupled with EIM these GSS APIs can be used to provide a Kerberos authentication mechanism for client server programs. This has the advantage that in house programs can be modified to implement single signon functionality. The APIs for the GSS can be found on the IBM Information Center by following these steps:

- Open a Web browser and enter the URL:

  http://publib.boulder.ibm.com/iseries/v5r2/ic2924/index.htm

- From the links section on the left the following items can be clicked. **Programming -> APIs -> APIs by category**, from here click the **security** link. And then click the **Generic Security Server APIs** link.

9.9 EIM demo tool

The EIM Java APIs, provided by IBM, allow programmers to enable their own user application for EIM or to create a tool for managing EIM domain data. To show you how to exploit EIM Java APIs an EIM demo tool was written that demonstrates the use of the EIM Java APIs. The tool allows an administrator to connect to an EIM domain using a non-encrypted or an encrypted (SSL) session. All connection-related information is stored in a properties file and displayed on the main window. The tool lets you perform the following tasks:

- Managing EIM Identifier
- Managing aliases and descriptions
- Managing associations
- Displaying registry information (note that the tool does not provide management of registry information)
- Displaying current user authorities

The Tool was written using Sun's JDK Version 1.4.1. Note that this tool is not a fully tested application that handles all possible exceptions and situations. You may want to customize it to your own needs. A README file is included that describes how to install, configure, and use the tool.

The tool can be downloaded from the FTP site at:

ftp://www.redbooks.ibm.com/redbooks/SG246975

Part 4

Appendices

The following appendices provide specific information that may be useful to some, but not all readers. Included are brief topics about:

- Saving and restoring your configuration files
- Troubleshooting your Network Authentication and Enterprise Identity Mapping configuration
- Windows support tools
- Planning forms
- Independent software vendor products
- Java source output

A

Backup and recovery

In this appendix we provide information needed for backup and recovery of the single signon components.

Microsoft Active Directory

Our scenario uses Kerberos authentication in the Microsoft Active Directory of your Windows 2000 server. Backup and recovery of the Active Directory is therefore essential. For details refer to the *Microsoft Technet* whitepaper *Active Directory Disaster Recovery* at:

`http://www.microsoft.com/technet/prodtechnol/windows2000serv/technologies/activedirectory/support/adrecov.mspx`

This document also covers the Domain Controller replication, which prevents the Kerberos Network Authentication Service to be the single point of failure in the network. If you want to make your network more reliable and available you should consider setting up Domain Controller replication.

Objects on your iSeries system

Details about the iSeries objects used to store the information for Network Authentication Service and EIM are given here.

For details about how to backup and recover iSeries objects, refer to *Backup and Recovery*, SC41-5304.

Important: Correct object authorities of the Network Authentication Service objects and EIM objects are essential for your system and network security. To maintain them after the restore, it is not sufficient simply to restore the objects. You need to Restore User Profiles (RSTUSRPRF) prior to restoring the objects and to Restore Authorities (RSTAUT) after restoring the objects. For details refer to *Backup and Recovery*, SC41-5304.

The iSeries Network Authentication Service objects

All objects used by the Network Authentication Service on the iSeries server are located in the Integrated File System (IFS) directory:

```
/QIBM/UserData/OS400/NetworkAuthentication
```

We recommend that you backup all objects and sub-directories of this directory on the same schedule that you back up your iSeries security data.

The EIM domain on the iSeries LDAP directory server

The EIM domain is implemented in an LDAP directory. If you use the directory server integrated in OS/400, it stores information in the following locations:

- The database library (QUSRDIRDB by default), which contains the directory servers contents. The name of the library can be found in iSeries navigator **Network -> Servers -> TCP/IP -> Directory Server** -> **Properties**.
- The QDIRSRV2 library, which is used to store publishing information (not used by EIM).
- The QUSRSYS library, which stores various items in objects beginning with QGLD (specify QUSRSYS/QGLD* to save them).
- If you configure the directory server to log directory changes, a database library called QUSRDIRCL that the change log uses.

We recommend that you backup the libraries and objects listed above on the same schedule that you back up your iSeries security data. It may be necessary to *save-while-active* since the option to end the LDAP server may not available to you.

The configuration data is stored in the following directory:

```
/QIBM/UserData/OS400/Dirsrv/
```

You should also save the files in that directory whenever you change the LDAP configuration or apply PTFs.

Another approach, from an availability perspective, is to use LDAP replication. Through V5R2, the OS/400 directory server has the capability to replicate data between a master server and one or more read-only replica servers. You can find more on replication in the IBM Redbooks on LDAP:

- *Implementation and Practical Use of LDAP on the iSeries Server*, SG24-6193
- *Understanding LDAP*, SG24-4986
- *LDAP Implementation Cookbook*, SG24-5110

The iSeries EIM configuration

Except for the data in the EIM domain in the LDAP directory, EIM uses configuration information you enter when you run the EIM configuration wizard. This configuration data contains the URL of the LDAP server and the name of the parent DN (if any).

The EIM configuration data is saved automatically with your security data using the Save Security Data (SAVSECDTA) command.

You restore it simply by restoring the QSYS user profile object.

Sample CL program to save your data

Example A-1 is a sample CL program that will save your data that relates to Network Authentication Service, LDAP, and EIM.

Example: A-1

```
/****************************************************************************/
/* SAVSSOOBJ - Save SSO objects (and more)                                  */
/* Parm: Device to which the objects are saved,                             */
/*       i.e. DEV parameter for various SAV... commands                     */
/*                                                                          */
/* Saves the following:                                                     */
/* - OS/400 security data including EIM configuration in the QSYS *USRPRF   */
/* - LDAP configuration and Network Authentication Service objects from IFS */
/* - LDAP objects from QUSRSYS (their name start with QGLD)                 */
/* - All default LDAP libraries.                                            */
/*   You may need too change the following in the SAVLIB command:           */
/*   * Change the name of the LDAP database library, if you changed         */
/*     the default name in DIrectory Server Properties                     */
/*   * Remove the name of the QUSRDIRCL library if you have not set up      */
/*     the logging of changes.                                              */
/*   * Remove the name of the QDIRSRV2 library if you are not using         */
/*     directory publishing                                                 */
/****************************************************************************/
PGM        PARM(&DEV)
DCL        VAR(&DEV) TYPE(*CHAR) LEN(10)    /* PARM: Device for save commands */
/* Establish error handling */
```

```
MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(ERROR))
/* Save security data including EIM configuration in the QSYS *USRPRF */
SAVSECDTA  DEV(&DEV) ENDOPT(*LEAVE)
/* Save LDAP configuration and Network Authentication Service objects - IFS */
SAV        OBJ(('/QIBM/UserData/OS400/NetworkAuthentication/*') +
               ('/QIBM/UserData/OS400/DirSrv/*')) +
           DEV(('/QSYS.LIB/' *TCAT &DEV *TCAT '.DEVD'))  ENDOPT(*LEAVE)
/* Save LDAP objects from QUSRSYS */
SAVOBJ     OBJ(QGLD*) LIB(QUSRSYS) DEV(&DEV) ENDOPT(*LEAVE)
/* Save-while-active all LDAP libraries - not only those used by EIM */
SAVLIB     LIB(QUSRDIRDB QUSRDIRCL QDIRSRV2) SAVACT(*SYNCLIB) DEV(&DEV)
/* Normal return from the program */
RETURN
/* Error handling: message to the operator and to the job log, abnormal end */
ERROR:
SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) TOMSGQ(*SYSOPR) +
           MSGDTA('Error when saving SSO/EIM/LDAP configuration +
                   and data. See job log for details.') +
SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) MSGTYPE(*ESCAPE) +
           MSGDTA('Error when saving SSO/EIM/LDAP configuration +
                   and data. See job log for details.') +
ENDPGM
```

B

Troubleshooting

This appendix provides a short guide to troubleshooting Network Authentication Service and Enterprise Identity Mapping.

Common problems and solutions

EIM is composed of multiple technologies and many applications and functions. Because there are many paths that can be taken to troubleshoot problems, the following topics contain detailed information and instructions about how to troubleshoot or fix some of the common errors that you may encounter, such as:

- Unable to connect to domain controller
- List EIM identifiers takes a long time
- EIM Configuration wizard hangs during finish processing
- EIM handle is no longer valid
- Kerberos authentication and diagnostic messages

Unable to connect to domain controller

A number of factors can contribute to connection problems when trying to connect to the domain controller. Check the following items to help find the cause of the problem:

- Verify that the information specified for the following items are correct:
  - Domain controller name
  - Specified port
  - User ID and password
- Verify that the domain controller is active. If the domain controller is an iSeries server, you can use iSeries Navigator and follow these steps:
  - Expand **Network -> Servers** -> **TCP/IP**.
  - Verify that the Directory Server has a status of Started. If the server is stopped, right-click **Directory Server** and select **Start**.

Once the domain controller is active, try reconnecting to the domain.

To do this, expand **Network** -> **Enterprise Identity Mapping** -> **Domain Management**.

Select the domain to which you want to connect. If no EIM domains are listed or the EIM domain you want to work with is not listed under the Domain Management folder, you need to add an EIM domain to domain management. See 7.2.2, “Add the EIM domain to be managed” on page 116 for instructions on adding domains to the iSeries Navigator view.

Right-click the EIM domain to which you want to connect and select **Connect**.

Specify the user type and the required user information that should be used to connect to the EIM domain controller, then click **OK**.

List EIM identifiers takes a long time

When opening the Identifiers folder in iSeries Navigator, it may take a long time for the list of identifiers to be generated. You may want to narrow the search criteria for displaying the list of EIM identifiers if you have a large number of EIM identifiers in your domain.

To customize the view for EIM identifiers, follow these steps:

- In iSeries Navigator, expand **Network -> Enterprise Identity Mapping** -> **Domain Management**.
- Expand the domain in which you want to display the EIM identifiers.
- Right-click **Identifiers** and select **Customize this view** -> **Include**.

- Specify the display criteria that you want. The asterisk (*) character may be used as a wildcard character.
- Click **OK**.

The next time you click **Identifiers**, the EIM identifiers displayed are only those that match the criteria that you specified. If you want to view all EIM identifiers, use the steps above and select All Identifiers as your customized view option.

EIM Configuration wizard hangs during finish processing

If the wizard appears to hang during finish processing, the wizard may be waiting for the domain controller to start. Verify that no errors occurred during the startup of the LDAP server. For iSeries servers, check the job log for the QDIRSRV job in the QSYSWRK subsystem.

To check the job log, follow these steps:

- In iSeries Navigator, expand **Work Management** -> **Subsystems -> Qsyswrk**.
- Right-click **Qdirsrv** and select **Job Log**.

EIM handle is no longer valid

While managing EIM through iSeries Navigator, if the user receives an error indicating that the EIM handle is no longer valid, the connection to the domain controller has been lost.

To reconnect to the domain controller, follow these steps:

- In iSeries Navigator, expand **Network -> Enterprise Identity Mapping ->Domain Management**.
- Right-click the domain that you want to work with and select **Reconnect**.
- Specify the connection information.
- Click OK.

Cannot connect with NetServer

After configuring Netserver your Window 2000/XP client get a message stating they are not authorized. You also see a message CPD3E3F with rc X'00000004' which points to wrong user mappings in the QZSOSIGN and QZLSERVER joblog. This may indicate that you did not map your Windows users through EIM prior to enabling NetServer to use Kerberos. See the text in 8.3.1, “Before you begin” on page 146. If this is the case you will either need to map your users or delete the Active Directory principal ID to get back to all users being prompted.

Kerberos authentication and diagnostic messages

The following are errors that can be found either in the joblog on the iSeries or displayed on the client when attempting to connect using Kerberos tickets.

CPD3E3F Network Authentication Service error

When using the Kerberos protocol for authentication with EIM, diagnostic message CPD3E3F is written to the job log whenever the authentication or identity mapping operations fail. The diagnostic message contains both major and minor status codes to indicate where the problem occurred. The most common errors are documented in the message along with the recovery.

See members KRB5 and GSSAPI in file H in library QSYSINC for the meaning of the major and minor status codes not listed in this message. The library QSYSINC is loaded under Option 13 of the operating system.

CWBSY1012 - Kerberos principal not found on server

Either the KDC did not have the principal entry for the iSeries, or we are asking incorrectly. In this case, the client was resolving the hostname for the iSeries incorrectly.

CWBSY1017 - rc=608 Kerberos credentials not valid on server

The iSeries apparently did not think the ticket received was intended for its service. In this case, the iSeries was not resolving its own host name correctly. See 2.2.1, “General TCP/IP considerations” on page 19 for further information on how to resolve this problem.

CWBSY1017 - rc=612 Kerberos credentials not valid on server

Again the iSeries did not like the ticket received. In this case, the problem was that the password for the secret key entered on the KDC did not match the password provided when running the Network Authentication Service wizard that created the key.

CWBSY1018 - Kerberos credentials could not be mapped to user

The Kerberos authentication was successful, but the Windows user was mapped incorrectly in EIM.

Errors when running client commands in QSH

When testing your Network Authentication Service setup you may encounter the following errors in the Qshell environment.

EUVF06014E Unable to obtain initial credentials

- Status 0x96c73a9c - Unable to contact security server. Make sure that the KDC is started and listening on the port you have configured in the iSeries krb5.conf file.
- Status 0x96c73a25 - Time differential exceeds maximum clock skew. The time setting on the KDC differs more than the skew allowed in the configuration files. Check the system values QTIME and QUTCOFFSET. Compare these values to the KDC’s time and offset from Greenwich mean time.
- Status 0x96c73ab5 - Key table entry is not found. Check the case of the fully qualified domain name and the principal name you are using, there is a problem resolving the name.
- Status 0x96c73a06 - Client principal is not found in security registry. There is a spelling error in the principal name or the fully qualified domain name. The client principal is unknown.
- Status 0x96c73a9a - Unable to locate security server. The realm name has is either misspelled or you have not used the correct case.

iSeries Access Diagnostic Tools

If you are receiving errors when working with iSeries Navigator you may be able to capture the errors using iSeries Access for Windows Diagnostic Tools. To start the tools on the PC, click **Start -> Programs -> IBM iSeries Access for Windows -> Service -> Start Diagnostic Tools**. An icon will appear in the tool tray indicating that the tools are active. Right-click the new icon and you will see that you have the choice to generate several types of traces including a history log, a detail trace, and an entry point trace. We found that the history trace was very helpful in debugging configuration errors in Network Authentication Service when performing iSeries Navigator functions.

Troubleshooting WebSphere Host On-Demand

The URL for WebSphere Host on Demand Information Center is:

http://www-306.ibm.com/software/webservers/hostondemand/library/v8infocenter/hod/en/help/2tabcontents.html

Support for Host on Demand can be found at the following URL:

http://www-306.ibm.com/software/webservers/hostondemand/library/v8infocenter/hod/en/tutorials/webexpress/troubleshooting.html

C

Windows 2000 Kerberos tools

Some programs included on the Windows operating systems CD-ROMs are not actually included in the installation of the operating system. These are usually tools to help perform specialized functions and are administrative tools which are installed only when they are needed. These actions can be used to perform all types of operations spanning from command line tools to help with Kerberos interoperability between operating systems, right through to GUI tools to restore deleted partitions on hard drives.

Introduction

This appendix details the installation of the support tools located on the Windows 2000 server CD-ROM and also how to verify that the Kerberos commands have been installed correctly.

Support tools installation

1. Place the Windows server cd into the CD-ROM drive on the server.
2. Navigate to the CDROM\Support\Tools directory.
3. The window you see should look similar to Figure C-1.

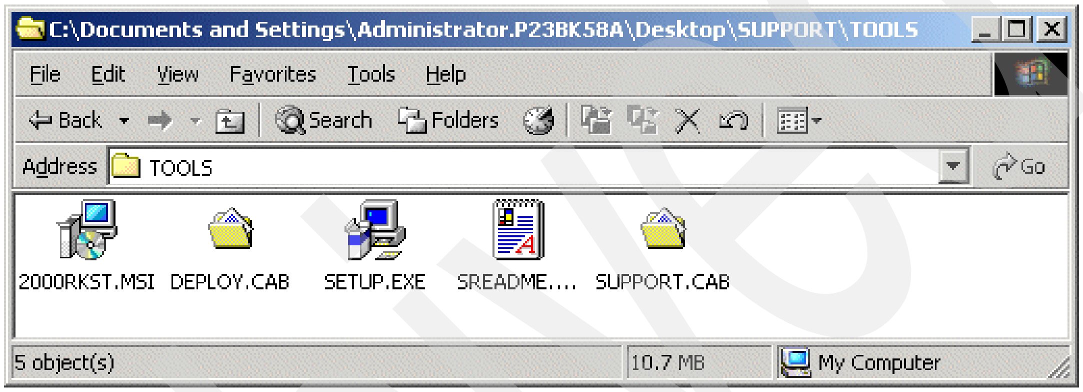


Figure C-1 Windows 2000 Server Support Tools Directory

4. Double-click the setup.exe and follow the default installation instructions until the support tools are installed.

Support tools verification

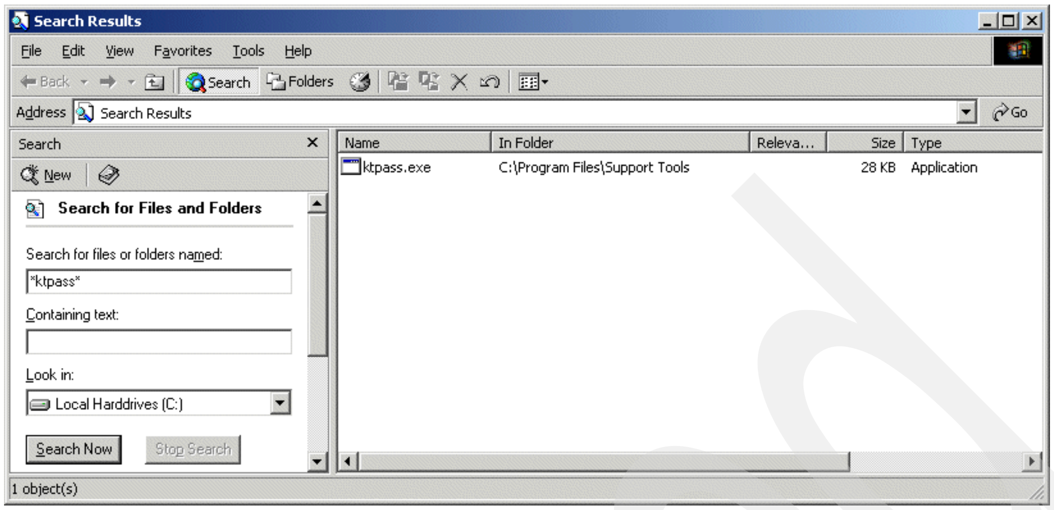
This section is about verifying that the support tools have been installed correctly on the Windows 2000 server.

Finding the ktpass command

The ktpass command can be found using a Windows file search.

1. Click the **Start** button in the bottom left-hand corner of the window.
2. Navigate to **Search** and then onto **Files and Folders**.
3. In the **Search for files or folders named:** box, enter `*ktpass*`.
4. Make sure that the **Look in:** field holds the value of the drive where the operating system is installed.
5. Click **Search Now**.

The computer will now search the file system for any files which have ktpass anywhere in the filename. The search results should return a reference to the ktpass command stored in Figure C-2.


Search Results
File Edit View Favorites Tools Help
Back Search Folders
Address Search Results Go
Search
New
Search for Files and Folders
Search Now Stop Search
1 object(s)

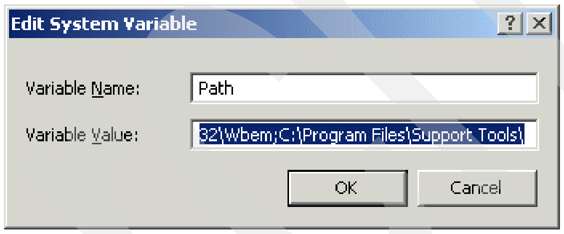
Search for files or folders named: *ktpass*
Containing text:
Look in: Local Harddrives (C:)

Name	In Folder	Releva...	Size	Type
ktpass.exe	C:\Program Files\Support Tools		28 KB	Application

Figure C-2 Finding the ktpass command

Verify the system path

1. Right-click **My Computer** and then click **Properties**.
2. Click the **Advanced** tab at the top of the window.
3. Then click the **Environmental Variables** button.
4. In the **System Variables** scroll to the **Path** variable and click **Edit**.
5. There should be the value of **`C:\Program Files\Support Tools\`** as in Figure C-3.



Variable Name: Path
Variable Value: 32\Wbem;C:\Program Files\Support Tools\

Figure C-3 The value of the path variable

Running the ktpass command

1. Click **Start** -> **Accessories** -> **Command Prompt The** command prompt window will appear.
2. From here you can run the ktpass command.

Attention: If after Step 3 has been completed a message is displayed indicating that the command cannot be recognized, then although the relevant information has been entered into the operating system it has not yet been saved. In order to save the information, the server can either be restarted or the logged on user can sign off and re-log onto the system. The ktpass command can then be executed from any path. Alternatively you can navigate to the c:\Program Files\Support Tools directory and run the command from there.

The ktpass command is used to provide interopability between windows and Unix style computers. The ksetup command is used to modify the windows registry to enable a different version of Kerberos other than the default Kerberos installed with the Windows server. The ksetup command will be installed with the ktpass command to verify its installation and if it will run follow the steps in the Verification section of this Appendix replacing a reference to ktpass with a reference to ksteup.

Klist command

The klist command does not come as standard on Windows machines and is a very useful debugging tool. The klist command can be used at various stages of program access to list the tickets which are applicable to this user. To download the klist program, visit the link below. The klist command is also run from the command prompt. The default installation directory is C:\Program Files\Resource Kit\ but the command list the ktpass command can be run from any directory at the command prompt.

`http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/klist-o.asp`

Kerbtray

Kerbtray is a graphical tools which sits in the system tray on a user desktop. This application contains useful tools such as the ability to look at the tickets which the user has on their computer and the remaining lifetime of these tickets. It also provided notifications of when tickets are due to expire on a system and when they will get renewed by the local systems Security Authority (LSA).

`http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/kerbtray-o.asp`

D

Planning forms

This appendix provides the planning forms referenced throughout this book. To obtain the best results you should fill them in prior to beginning the configuration of each of the components.

Important: Any and all passwords specified in this book are for example purpose only. To prevent a compromise to your system or network security, you should never use these passwords as part of your configuration.

Prerequisites checklist

Table D-1 provides a checklist to determine if you are ready to proceed with the configuration of Network Authentication Service and EIM.

Table D-1 Prerequisites checklist

Prerequisites	
Is your OS/400 V5R2 (5722SS1) or later? (required for this scenario)	
Are the following options and licensed products installed on your iSeries server? ► OS/400 Host Servers (5722SS1 Option 12) ► Qshell Interpreter (5722SS1 Option 30) ► iSeries Access for Widows (5722XE1) ► Cryptographic Access Provider (5722AC3)	
Is iSeries Access for Windows (5722XE1) installed on all the PC's that are to be used for single signon?	
Is iSeries Navigator installed on administrator's PC?	
Are the following two subcomponents of iSeries Navigator included on the administrator's PC? ► Security subcomponent ► Network subcomponent	
Have you applied the latest program temporary fixes (PTFs)?	
Have you installed the latest iSeries Access service pack?	
Do you, the administrator, have *SECADM, *ALLOBJ, and *IOSYSCFG special authorities?	
Do you currently have a Key Distribution Center (KDC) operating?	
If the KDC is to run on Windows (R) 2000 Server: Do you have Windows Support Tools installed on the system being used as the KDC? See Appendix C, "Windows 2000 Kerberos tools" on page 235.	
Are all your PCs in your network configured in a Windows (R) 2000 domain?	
Is the iSeries system time within 5 minutes of the KDC's system time? Refer to 2.2.2, "Time / SNTP" on page 21 on how to synchronize the system time.	
If an LDAP server is currently configured, can it be stopped temporarily? (This will be required to complete the EIM configuration process.)	
If you intend to use SSL to secure communications from the workstations (recommended): Is Client Encryption 128-bit (5722CE3) installed on your iSeries system and on all PCs from which you want the communication secure using SSL?	

Configuration planning worksheets

Instructions for completing this worksheet are located in 2.4, “Information to collect before you start” on page 24.

Table D-2 Planning worksheet

Item	Information to collect	Result
A	What is the name of the Kerberos default realm to which the iSeries will belong?	
B	What is the KDC for this Kerberos default realm?	
C	What is your KDC’s fully qualified host name?	
D	What is the port on which the KDC listens?	
E	What is name of the password server for this KDC?	
F	What is the port of your password server?	
G	What is the password for your iSeries service principal(s)?	
The following items will be used to create the iSeries principal on the KDC		
H	What is the name of the Kerberos principal?	krbsvr400 (when creating the iSeries principal this name must be used)
I	What is your iSeries host name?	
J	What is the fully qualified host name of the iSeries?	
K	What is the name of the Kerberos default realm to which the iSeries server belongs? (Default: domain name converted to upper case)	
L	What is the full name of the principal? (`krbsvr400/fully.qualified.host.name@YOUR.KERBEROS.REALM`)	
M	What is the password / shared secret for this principal? (Must be the same as item G)	
The following items will be used to configure Enterprise Identity Mapping (EIM)		
N	Which type of basic EIM configuration do you want to create on your iSeries system? ▸ Join an existing domain ▸ Create and join new domain	
O	Where do you want to configure your EIM domain, or what EIM domain you want to join?	
P	What is the name of the EIM domain you want to create or join?	
Q	Do you want to specify a parent DN for the EIM domain? If yes - specify the parent DN	

Item	Information to collect	Result
R	What is the administrator distinguished name (DN) on the LDAP server which will be used as the EIM domain controller?	
S	What is the administrator password on the LDAP server will be used as the EIM domain controller?	

Available EIM products

We are aware that the process of implementing an EIM environment does not actually create a single signon solution, only the infrastructure for one. There is still the task of entering information into the EIM domain. The high volume of data entry that could be required as a result of the number of users and servers could be quite daunting, especially for a large organization. In this section we describe the tools which are available to help you fulfil this last part of the EIM infrastructure.

BlueNotes EIM Administration Suite

The BlueNotes EIM Administration Suite is designed to provide a simple solution to addressing your implementation of EIM. The administration suite consists of the following products:

- Directory Synchronization for OS/400
- DataPump
- EIM Administrator

The suite provides an effective method of addressing the three different steps referred to in 6.5, "Three steps to success" on page 90 for the successful implementation of EIM. These three steps are:

- The collection of all your user IDs.
- The matching of these IDs with the relevant users.
- Incorporating these into your EIM infrastructure.

To download an evaluation copy of BlueNotes EIM Administrator or for more information about the suite, visit:

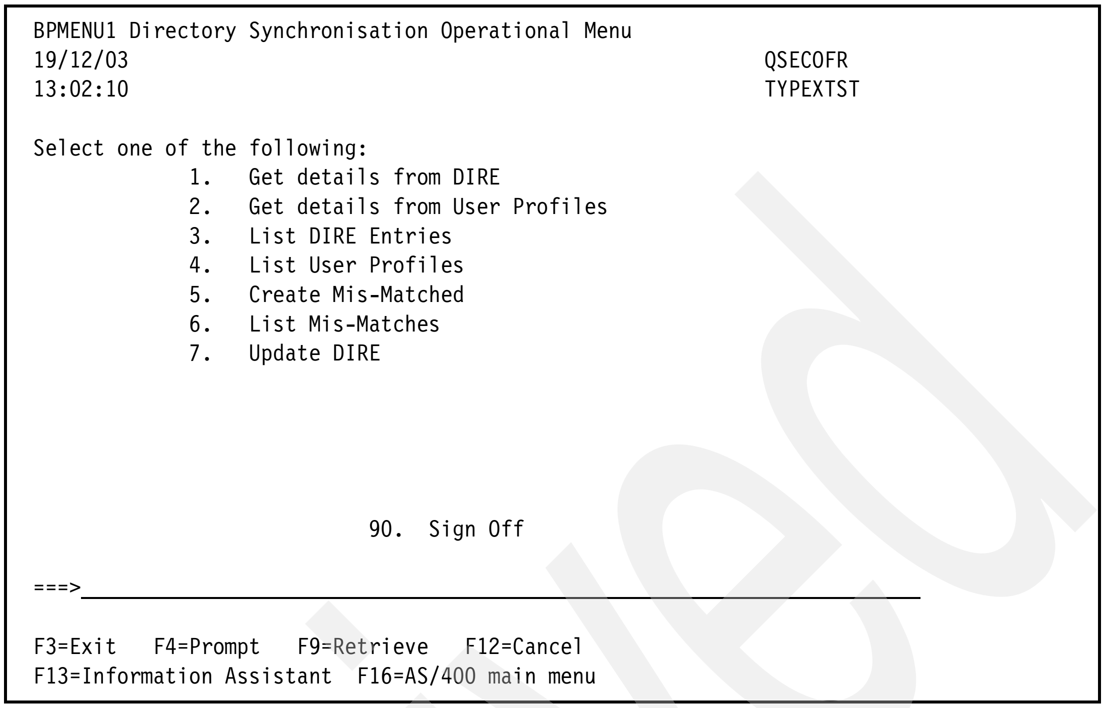
`http://www.bluenotes.com`

Overview

The first step in implementing EIM is the collection phase. Here you need to identify all of your users prior to inputting them into EIM. However, the main issues at this stage are not only the number of different systems where user IDs exist but also the number of user IDs on those systems themselves.

To identify users for the collection phase, BlueNotes Directory Synchronization offers an automated interface between OS/400 user profiles, OS/400 System Distribution Directory and an LDAP enabled directory. This could be an IBM Directory Server, a Domino Server or Microsoft's Active Directory. IBM's Directory Integrator is an alternative tool to help you collect user information from systems other that OS/400.

Figure E-1 shows the user interface for BlueNotes Directory Synchronization.



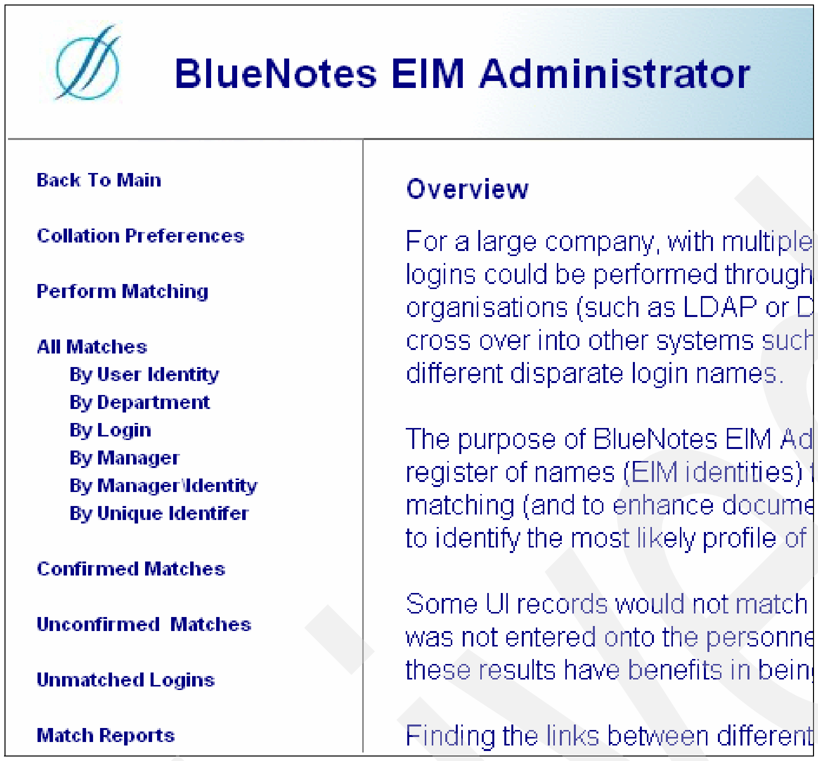
```
BPMENU1 Directory Synchronisation Operational Menu
19/12/03                                                QSECOFR
13:02:10                                                TYPEXTST

Select one of the following:
             1.   Get details from DIRE
             2.   Get details from User Profiles
             3.   List DIRE Entries
             4.   List User Profiles
             5.   Create Mis-Matched
             6.   List Mis-Matches
             7.   Update DIRE




                           90.  Sign Off

===>______________________________________________________________________

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel
F13=Information Assistant  F16=AS/400 main menu
```

Figure E-1 BlueNotes Directory Synchronisation for OS/400

Collection and collation

The next tool in the suite is BlueNotes EIM Administrator which enables you to match all the user IDs to the users in your organization and use that information to populate the EIM domain. BlueNotes EIM Administrator has two functions, collation and population.

Figure E-2 shows the menu that the user will see when the collation section of BlueNotes EIM Administrator is opened. It shows the collation preferences and configuration information sections. The *Perform Matching* button starts a process which links the list of user names shown in Figure E-3 with the master list of users shown in Figure E-4. Additionally, it shows the different ways information can be presented in the application after the matching process has been performed.


BlueNotes EIM Administrator

Back To Main

Collation Preferences

Perform Matching

All Matches
- By User Identity
- By Department
- By Login
- By Manager
- By Manager\Identity
- By Unique Identifer

Confirmed Matches

Unconfirmed Matches

Unmatched Logins

Match Reports

Overview

For a large company, with multiple
logins could be performed through
organisations (such as LDAP or D
cross over into other systems such
different disparate login names.

The purpose of BlueNotes EIM Ad
register of names (EIM identities) t
matching (and to enhance docume
to identify the most likely profile of

Some UI records would not match
was not entered onto the personne
these results have benefits in bein

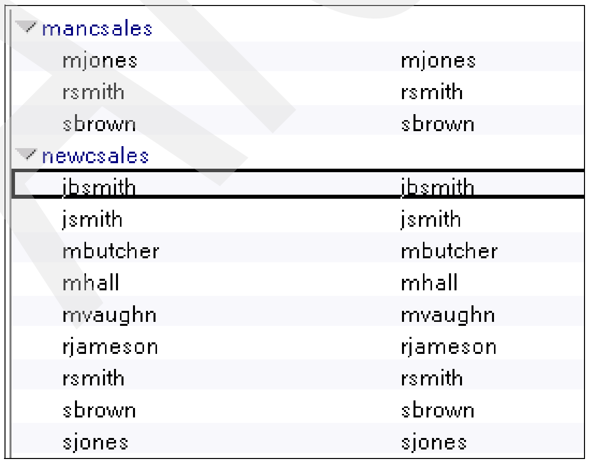
Finding the links between different

Figure E-2 Collation section

To simplify the matching process a master list is used. A master list is a platform to start from when comparing the collected user names. Ideally this should be a list which has a unique value for each user, such as an employee number. This information could be collected using, the third tool, BlueNotes DataPump, which provides an interface between a variety of ODBC sources and a Lotus Notes database.

This unique information could be directly input into BlueNotes EIM Administrator ready for the matching and population process. EIM Administrator takes all the collected user IDs and, using a collation process, maps these user IDs to relevant users.

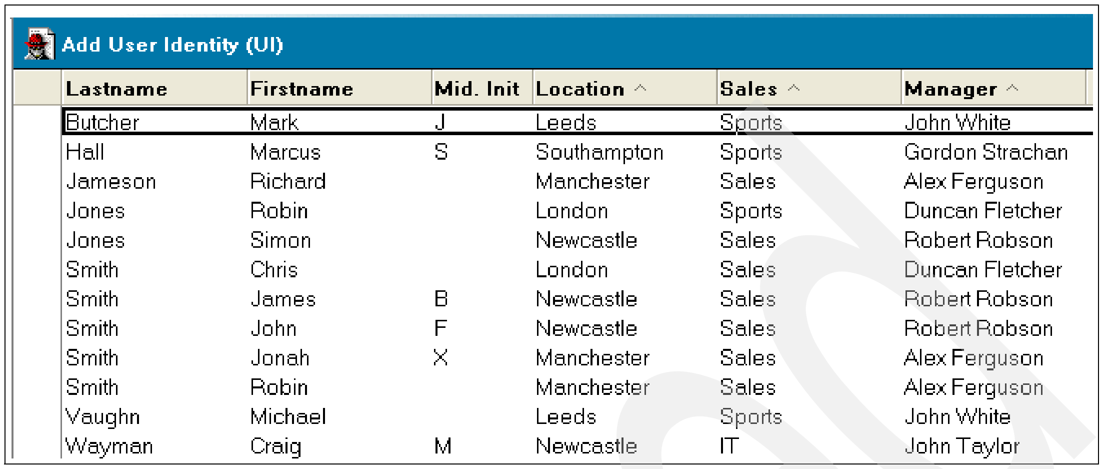
Figure E-3 shows a list of collected user IDs which are ready for the matching process.



mancsales	
mjones	mjones
rsmith	rsmith
sbrown	sbrown
newcsales	
jbsmith	jbsmith
jsmith	jsmith
mbutcher	mbutcher
mhall	mhall
mvaughn	mvaughn
rjameson	rjameson
rsmith	rsmith
sbrown	sbrown
sjones	sjones

Figure E-3 Collect user IDs relating to the mancsales and newcsales servers

Figure E-4 shows the master list which is held in BlueNotes EIM Administrator. The inclusion of a master list makes the collation process much more efficient and a lot simpler to perform.



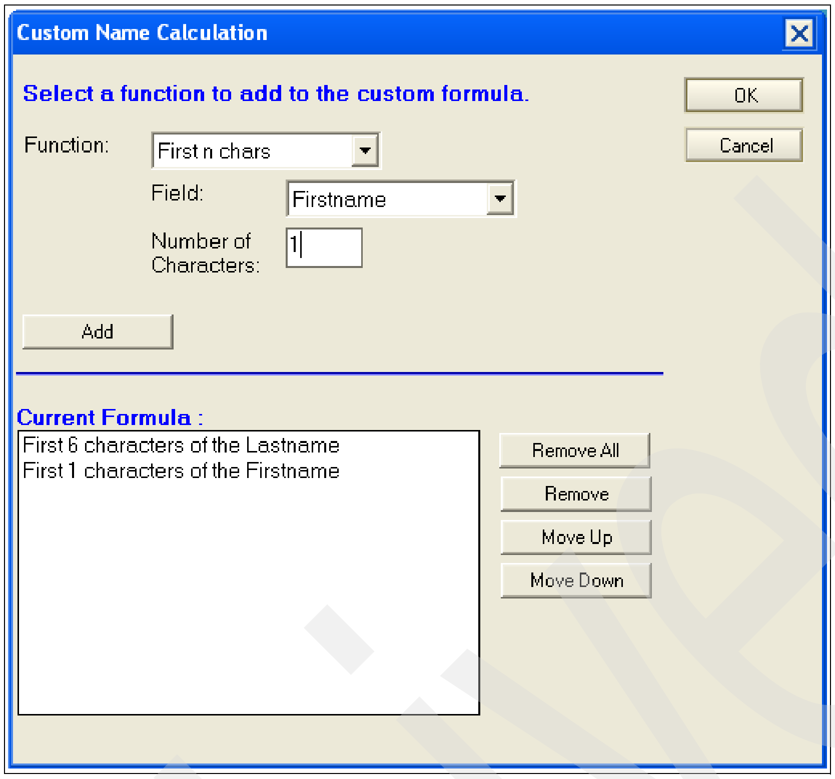
Add User Identity (UI)

Lastname	Firstname	Mid. Init	Location	Sales	Manager
Butcher	Mark	J	Leeds	Sports	John White
Hall	Marcus	S	Southampton	Sports	Gordon Strachan
Jameson	Richard		Manchester	Sales	Alex Ferguson
Jones	Robin		London	Sports	Duncan Fletcher
Jones	Simon		Newcastle	Sales	Robert Robson
Smith	Chris		London	Sales	Duncan Fletcher
Smith	James	B	Newcastle	Sales	Robert Robson
Smith	John	F	Newcastle	Sales	Robert Robson
Smith	Jonah	X	Manchester	Sales	Alex Ferguson
Smith	Robin		Manchester	Sales	Alex Ferguson
Vaughn	Michael		Leeds	Sports	John White
Wayman	Craig	M	Newcastle	IT	John Taylor

Figure E-4 The master list of user IDs held in BlueNotes EIM Administrator

The matching process uses information about each user, such as their location and manager, to generate an increasing probability of a match between a user ID and a user. These probabilities can be personalized for your organization. In addition to holding information about each user, a taxonomy module is also included to improve the probability of a match. For example, if a user is in Rochester but the server that he accesses is defined as being in Minnesota, the taxonomy database can be set to know that Rochester is in Minnesota and the probability therefore reflects a higher match than if the users location and the server location did not match.

In addition to the matching process described above, there is another level on which BlueNotes EIM Administrator performs matching. When a list of servers is input to EIM Administrator a selection is made to determine the rule for that server on how user names are usually generated. This can either be from a pre-selectable list or can be customized. BlueNotes EIM Administrator provides a very effective interface for customized user name calculations. More than one user name generation formula can be created allowing for situations where users with duplicate names will have different IDs. For example CWayman, and CWayman2 are perfectly valid IDs but represent different users. Figure E-5 shows the dialog box which is used to create custom name calculations.



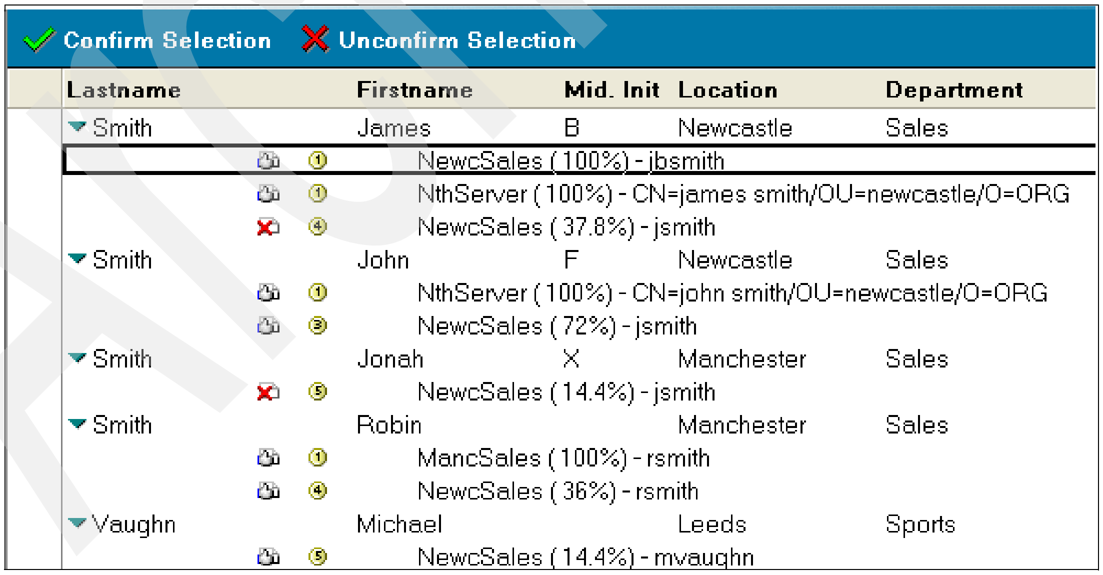
Function: First n chars
Field: Firstname
Number of Characters: 1

Current Formula :

First 6 characters of the Lastname
First 1 characters of the Firstname

Figure E-5   Custom user name calculation dialog box

After the matching process has completed, users can view a list of the users and user names associated to them by the matching process. These assigned user IDs can then be confirmed. This confirmation flags the entry as being ready for population into EIM.



Lastname	Firstname	Mid. Init	Location	Department
Smith	James	B	Newcastle	Sales
	NewcSales (100%) - jbsmith			
	NthServer (100%) - CN=james smith/OU=newcastle/O=ORG			
	NewcSales (37.8%) - jsmith			
Smith	John	F	Newcastle	Sales
	NthServer (100%) - CN=john smith/OU=newcastle/O=ORG			
	NewcSales (72%) - jsmith			
Smith	Jonah	X	Manchester	Sales
	NewcSales (14.4%) - jsmith			
Smith	Robin		Manchester	Sales
	ManciSales (100%) - rsmith			
	NewcSales (36%) - rsmith			
Vaughn	Michael		Leeds	Sports
	NewcSales (14.4%) - mvaughn			

Figure E-6   View showing the matched documents after the collection process

These confirmed documents can then be shown in confirmed views separating out the unconfirmed matches. There is a third type of view to show lists of user IDs for which no matches could be found. Another function of BlueNotes EIM Administrator allows the manual assigning of particular user IDs to a user. Therefore the probability ranking of this action is regarded as a 100% match as it has been performed by a person who has the knowledge to verify this information. After the process of confirming valid user IDs has been completed you may choose to only confirm IDs for, say, a particular department. These IDs can then be populated automatically into the EIM domain.

This highlights one of the main advantages of EIM, that the user IDs for a whole organization do not have to be in the EIM domain for reduced signon benefits to be seen by users.

The matching process also produces a report to ensure it completed successfully and to give details of the actions that occurred during the matching process. This concludes the collation section of BlueNotes EIM Administrator.

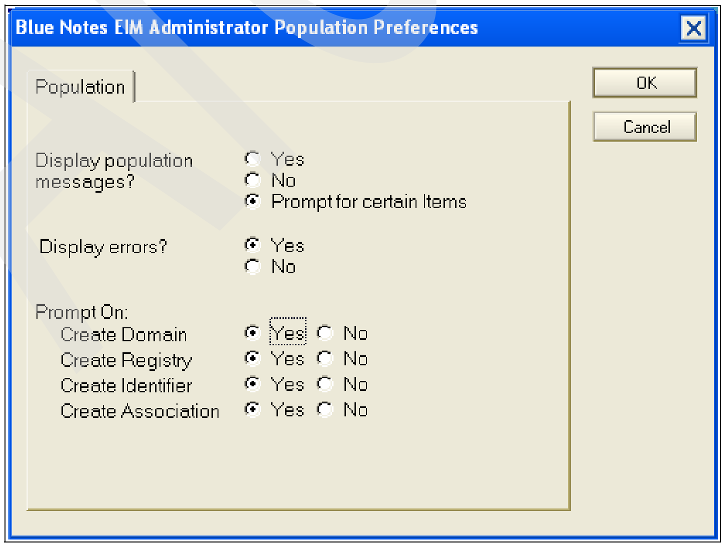
Population

We now move onto the population function. This allows the confirmed user IDs to automatically be input into the EIM domain. Population consists of a configuration section plus an optional conversation with the administrator while population is happening.

Figure E-7 shows the population configuration dialog box which user will see when the population configuration link is selected. These population options dictate the interaction between the user running the population task and the population task itself. This interaction between the user and the application is designed to instill confidence in the user that the population task is functioning correctly. When the user has run the population process with interaction, they then have the chance to run it without interaction. The administrator can then check the population log to ensure the process completed successfully.

The population configuration allows the user to:

- Define what messages are displayed.
- Decide if error messages should be displayed to the user in addition to being written to the log.
- Decide if choices should be presented to the user, such as:
  - All messages
  - No messages
  - Or messages for objects which the population process will need to create in order to fulfill this process such as domains, registries, identities and associations.



Display population messages? Yes No [✓ Prompt for certain Items]

Display errors? [✓ Yes] No

Prompt On:

Create Domain [✓ Yes] No

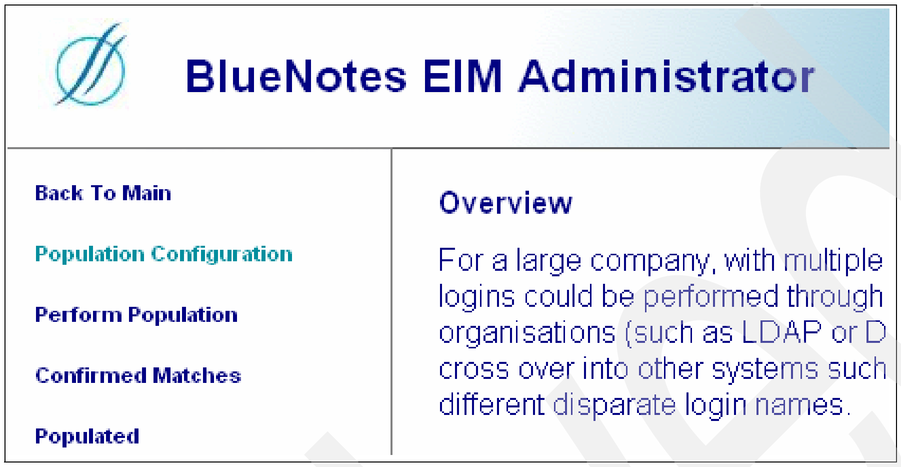
Create Registry [✓ Yes] No

Create Identifier [✓ Yes] No

Create Association [✓ Yes] No

Figure E-7 Population configuration

The links below show the population section of BlueNotes EIM Administrator and will present the options as described above. The *Perform Population* link starts the population process based on the confirmed user IDs from the collation process. The population process uses the EIM Java APIs to interact with the EIM domain. The populated view shows which entries have been moved into the EIM domain.



For a large company, with multiple
logins could be performed through
organisations (such as LDAP or D
cross over into other systems such
different disparate login names.

Figure E-8 Population section

Summary

In summary the BlueNotes Administration Suite is an effective software solution for gathering and inputting information into an EIM domain. It also has many benefits:

- Greater efficiency in creating the information in EIM.
- Greater ease in matching user IDs with real users.
- Increased security through removing redundant IDs.
- Accurate representation of the number of licenses actually needed for your employees.

To find out more about BlueNotes EIM Administrator, visit the Web site at:

`http://www.bluenotes.com`

Or, contact the Typex Group by e-mail at:

`info@typex.com`

Or, by telephone at +44 (0) 191 256 4402.

SafeStone’s AxcessIT - Automated EIM Management

The SafeStone AxcessIT suite of modules is a complete Identity and Access Management product. Within this product as part of the natural automation you are able to seamlessly populate and manage the EIM registry. This is an independent platform solution, built in JAVA and accessed via a Web browser that manages the EIM registry as part of a provisioning process.

The modules that interact with EIM are AxcessIT Provisioning Manager and AxcessIT Password Manager. Each of these products deals with users that have multiple target user-id associations. For the purpose of this redbook, we concentrate on the Provisioning Manager although the same concept is used for both (exception being Role management).

The main features that are utilized to manage the EIM registry within AxcessIT Provisioning Manager are:

- Orphaned Target Account Import -This allows for collection of target accounts from each of the managed systems and applications. These accounts are brought back into the AxcessIT data store for use in the registration process.
- Register Target Account with an AxcessIT User - The process of associating target accounts with an AxcessIT user account. Utilizes the results from the Orphaned Target Account import process. Invokes the EIM registry update process.
- Role Assignment - Full provisioning process of creating user accounts on multiple targets and transparently updating and maintaining the EIM Registry.

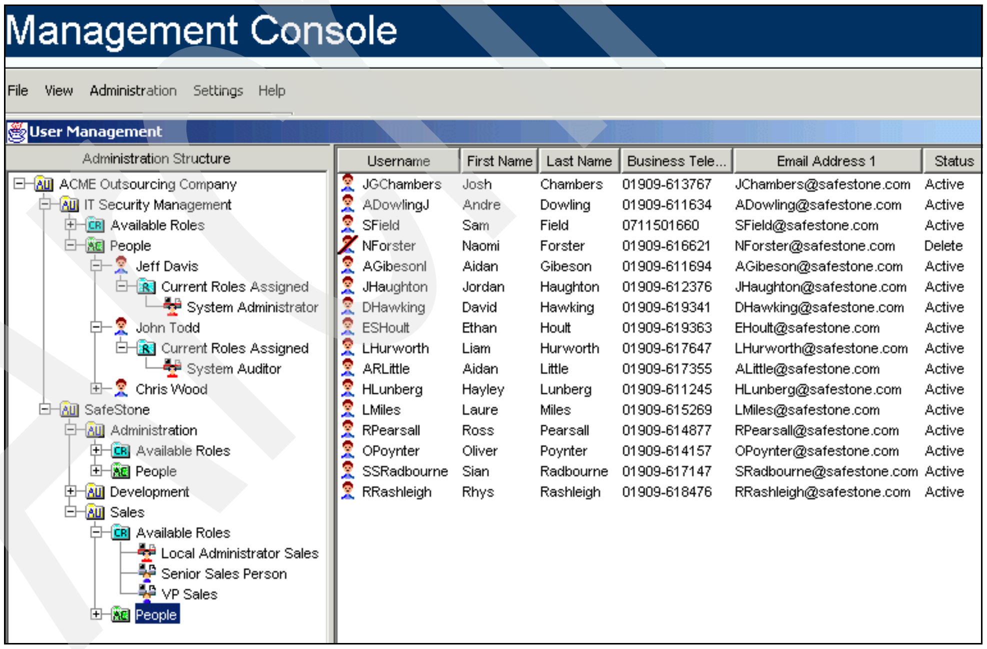
These three features address the steps outlined in 6.5, “Three steps to success” on page 90.

Overview

AxcessIT Provisioning Manager is a “tree” orientated management process. An Administration Tree is built that represents how the organization is managed from a user administration perspective (Figure E-9).

The tree is divided into various “Administration Units” or “AU’s”. There can be as many AU’s as required and the tree can be as deep and wide as required.

Within each AU AxcessIT users are defined. This can either be done manually or processed from a data store via an XML import process. These AxcessIT User Accounts become the focal point for all target account associations. Once all the users are defined in the relevant AU’s you can move onto the next step in the process.



Username	First Name	Last Name	Business Tele...	Email Address 1	Status
JGChambers	Josh	Chambers	01909-613767	JChambers@safestone.com	Active
ADowlingJ	Andre	Dowling	01909-611634	ADowling@safestone.com	Active
SField	Sam	Field	0711501660	SField@safestone.com	Active
NForster	Naomi	Forster	01909-616621	NForster@safestone.com	Delete
AGibesonI	Aidan	Gibeson	01909-611694	AGibeson@safestone.com	Active
JHaughton	Jordan	Haughton	01909-612376	JHaughton@safestone.com	Active
DHawking	David	Hawking	01909-619341	DHawking@safestone.com	Active
ESHoult	Ethan	Hoult	01909-619363	EHoult@safestone.com	Active
LHurworth	Liam	Hurworth	01909-617647	LHurworth@safestone.com	Active
ARLittle	Aidan	Little	01909-617355	ALittle@safestone.com	Active
HLunberg	Hayley	Lunberg	01909-611245	HLunberg@safestone.com	Active
LMiles	Laure	Miles	01909-615269	LMiles@safestone.com	Active
RPearsall	Ross	Pearsall	01909-614877	RPearsall@safestone.com	Active
OPoynter	Oliver	Poynter	01909-614157	OPoynter@safestone.com	Active
SSRadbourne	Sian	Radbourne	01909-617147	SRadbourne@safestone.com	Active
RRashleigh	Rhys	Rashleigh	01909-618476	RRashleigh@safestone.com	Active

Figure E-9 Administration Tree Structure – Master Console View

Orphaned Target Account processing

This is invoked by selecting a managed target within the AxcessIT Management Console, (*Figure E-10)*, right-clicking on the target to display the content sensitive menu and then choosing the Search for Orphaned Target Accounts item. This invokes the process via the AxcessIT communication adaptors and compiles a list of accounts that do not have an association with an AxcessIT user. After you register target accounts you can rerun this process to produce a decreasing list. The list also decreases in size as you “use up” the target accounts.

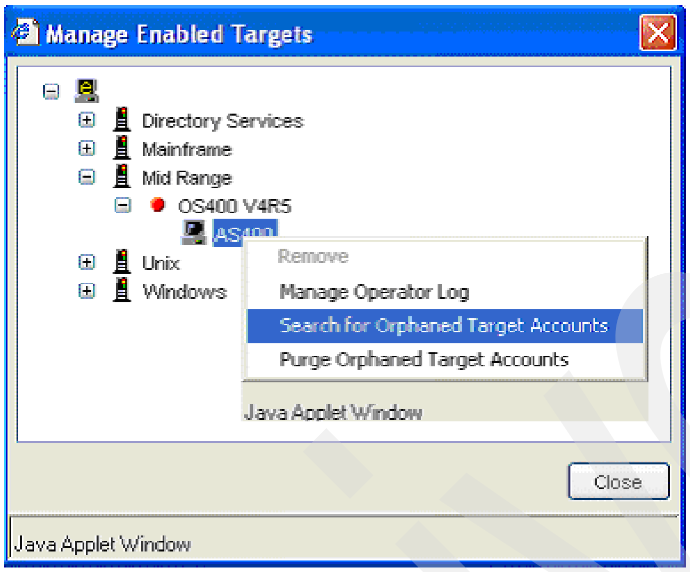


Figure E-10 Invocation of Orphan Target Account Process

Register Target Account processing

Association of target user-ids with an AxcessIT user is performed via the *Register Target Account* feature. However, before you can perform this process you must first obtain lists of target accounts on each of the managed targets. This is done via the Orphaned Target Account process as described above. An Orphaned Target Account is an account that has yet to be associated with an AxcessIT User. These are also known as rogue accounts – accounts set-up outside the AxcessIT Provisioning product on the local system rather than part of the managed process.

The next step in the process is to register these accounts with their owner – the AxcessIT user account.

Within the administration tree structure you select a user and, using the right-click, display the available menu items (Figure E-11). One of the items will be the ability to register a target account. By selecting this option a list of managed targets, those that have an Orphaned Account List produced, will be presented to the Administrator. By selecting one of these targets the list of Orphaned Accounts will be displayed. This has a filtered search facility built in so that, if already known, individual accounts can be displayed. If the account is not fully known a subset can be displayed or the full list.

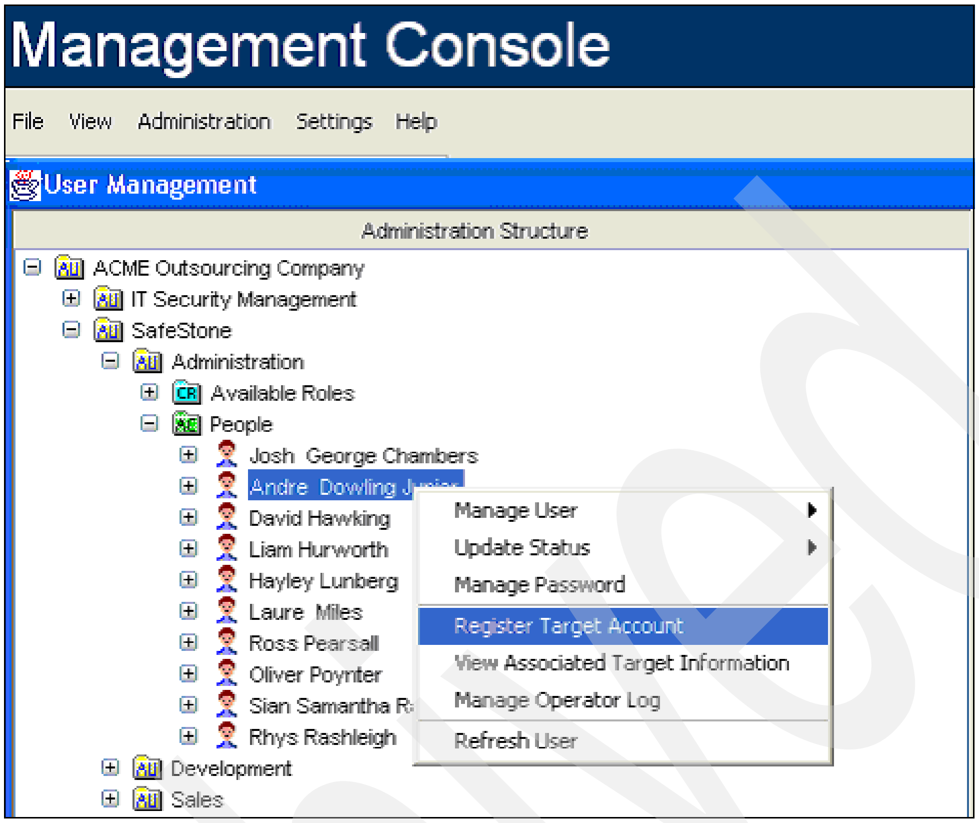
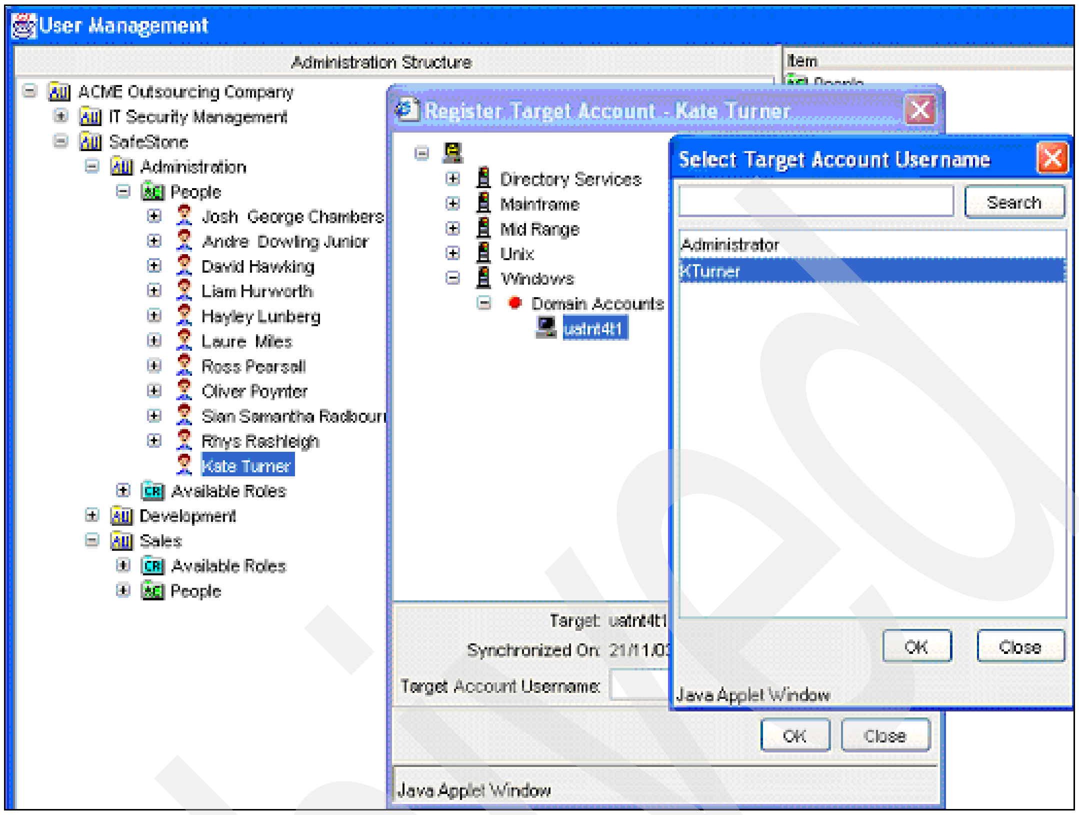


Figure E-11 Selecting menu item to invoke target account registration

During the registration process of targets – bringing a target online to AxcessIT, a default naming convention is specified. This is available to the “Register Target Account” process and therefore, when a list of Orphaned Target Accounts is presented for selection; the closest matching user-ids are placed at the top of the list. The administrator selects the relevant target accounts and confirms the association (Figure E-12).



Target: uatnt4t1

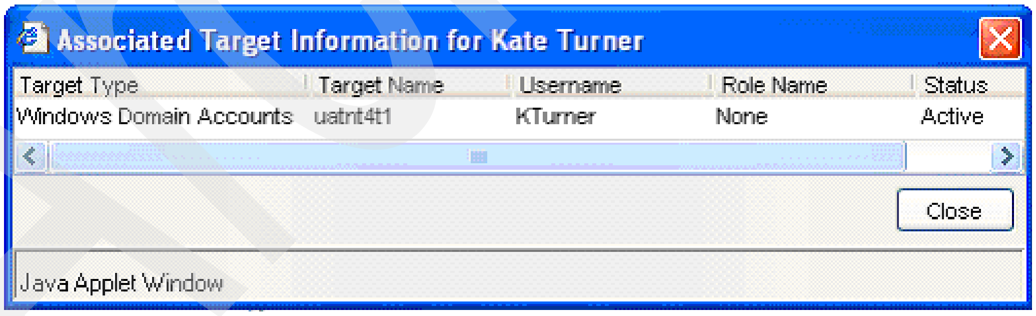
Synchronized On: 21/11/03

Target Account Username:

Figure E-12 Register target Account to a selected user

The Administrator can select one or more of the displayed accounts to build the association up. The process is repeated until all Orphaned Accounts have been accounted for.

By viewing the associated target accounts of an AxcessIT User you can obtain a full list of all accounts across all targets for that user (Figure E-13).



Target Type	Target Name	Username	Role Name	Status
Windows Domain Accounts	uatnt4t1	KTurner	None	Active

Figure E-13 Associated Target Information for a selected user

These associations do not have to be a manual task, especially if a repository exists which has these associations already made. The “Import” facility can create these target account associations by specifying the accounts in the XML input document.

Population process

This now leaves the population of the EIM registry step to complete the process. AxcessIT Provisioning Manager is a role-based system. The population process is completely transparent as it is either invoked via a role assignment or an account registration (or the reverse) event. Roles are defined that allow user accounts to be set up on a multitude of targets in a single drag and drop operation or via a Workflow process. The EIM registry is seen as an additional target for AxcessIT Provisioning Manager to manage.

Technical overview

To create an EIM target, a registered target can be enabled as an EIM controller (Figure E-14). Other targets in AxcessIT can then be enabled to allow them to be associated with an EIM controller. These targets can be added into the EIM controller's domain, meaning that changes to these targets will be replicated on the EIM controller.

To create an EIM identifier on a controller, a role is created with the required EIM controller target. Upon assigning the role to a user an EIM identifier is created on the target. EIM administrator roles may also require assignment to AxcessIT users. This is achieved by creating a role with the EIM controller as a target and a connection to one of the EIM administrator groups. Multiple role assignment provides a convenient means to perform this aspect.

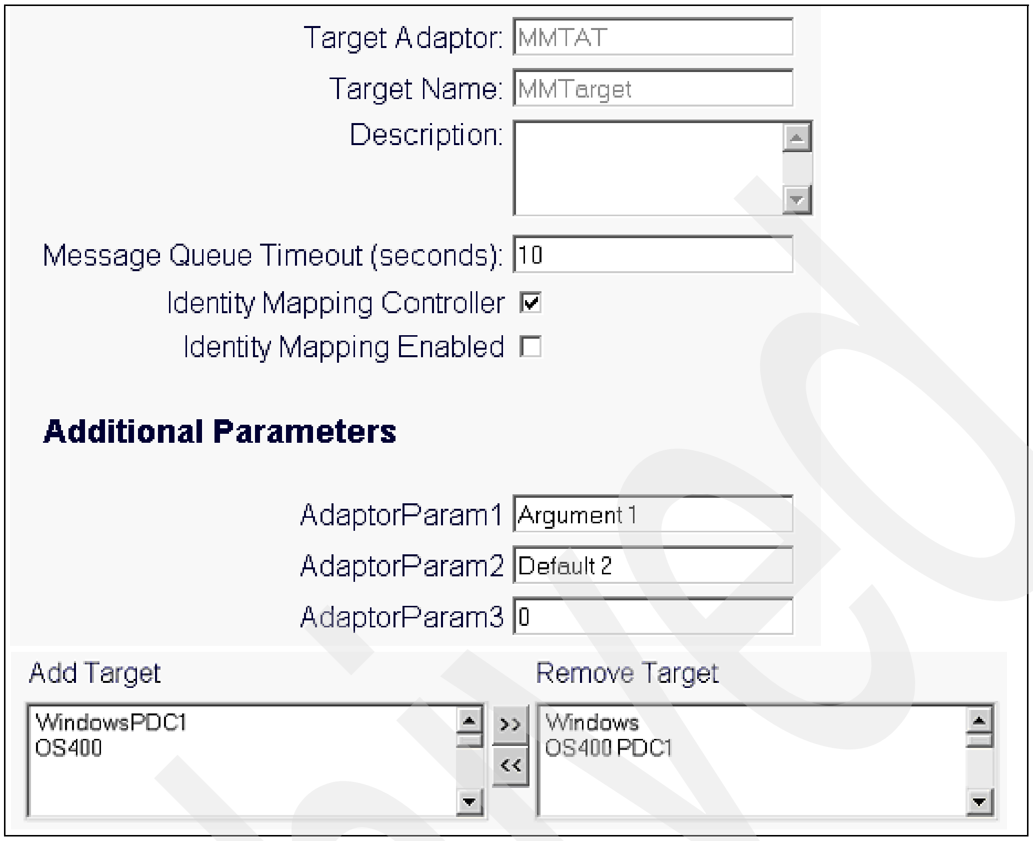
Once an AxcessIT user has an EIM identifier, the target account life cycle will be replicated against the identifier (that is, creation, modification and deletion).

Upon performing an assign role or register target account operation, the target in question will be queried for any EIM domain relationships. If any domain relationships are found the user is queried for a relating EIM target account. If an EIM target account is found, an update is performed on the EIM target account to add the newly created target account as a user registry entry.

Un-assign role and de-register target account operations, likewise, will perform updates upon the EIM identifier removing the user registry entry.

When un-assigning a role from a user the related EIM identifier is removed from the domain along with the related user registry entries.

This mechanism allows EIM identifiers and user registry entries to be created on EIM domains.



Target Adaptor:	MMTAT
Target Name:	MMTarget
Description:	
Message Queue Timeout (seconds):	10
Identity Mapping Controller	✓
Identity Mapping Enabled	☐

Additional Parameters

AdaptorParam1	Argument 1
AdaptorParam2	Default 2
AdaptorParam3	0

Add Target	Remove Target
WindowsPDC1	Windows
OS400	OS400 PDC1

Figure E-14 Target registration with AxcessIT – Showing EIM associations

The AxcessIT solution fully supports delegated administration by use of system roles. Local Administrators can be placed anywhere in the Administration Tree structure. The registration of users can then be delegated to each department in order to speed up the initial deployment process.

In summary the SafeStone AxcessIT suite of modules embraces the EIM Domain completely, managing the Identity Mapping process transparently as part of the Secure Identity and Access Management process.

To find out more about the SafeStone product visit the Web site:

`http://www.safestone.com`

You can also e-mail:

mailto:info@safestone.com

TriAWorks Identity Manager for Single Sign-On

TriAWorks Identity Manager for Single Sign-On (TIM SSO) is a product designed specifically to manage Enterprise Identity Mapping (EIM) domains. TIM SSO makes it easy for a security administrator to quickly and accurately create and maintain associations between a unique person and different user identities on registries across the enterprise.

In particular, TIM SSO addresses the population and collation steps discussed in 6.5, “Three steps to success” on page 90.

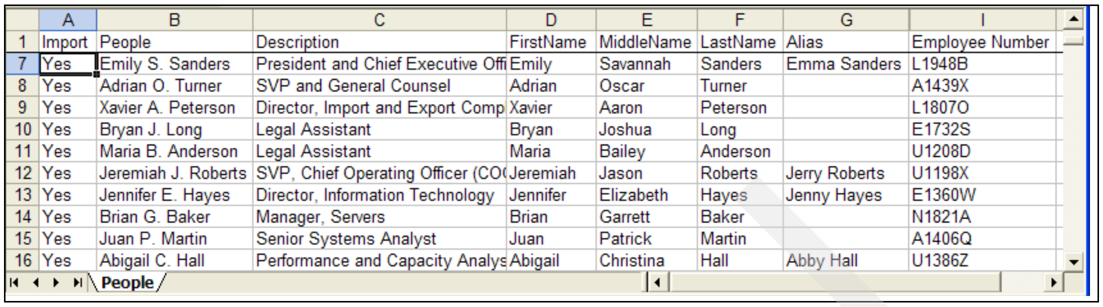
TIM SSO provides the following features to simplify EIM management and enable SSO:

- The Graphical User Interface (GUI) is logically organized into views that display an organization's People (EIM identifiers), Registries, User Identities, Associations, Suggestions, and the relationships between them.
- Simple and flexible wizards allow you to import data for the people, servers, and user identities. User identities can be loaded directly from properly configured LDAP servers.
- Find suggestions, which are user identities across the enterprise, that are potential associations for people. Suggestions are based on a set of customizable user identity composition rules. This significantly reduces the administrator burden to map user identities by eliminating the need to manually search numerous servers to identify potential associations.
- Easily and intuitively create associations using drag-and-drop from either the User Identities view or Suggestions view to the Associations view. Straightforward and powerful wizards may also be used to create associations.
- Rules are evaluated and tested against a selected person and the results displayed to ensure rules are constructed as expected before being used to find suggestions and create associations.
- Icons provide visual cues to potential warning conditions which may require quick correction, such as: (1) A user identity is mapped to more than one EIM identifier or person; or (2) an association exists with a non-existent user identity.
- Several people, user identity, and registry reports are available. Included are integrity reports that highlight the warning conditions described above.
- A dashboard quickly shows how many user identities have been mapped for a selected server. The dashboard helps quickly determine how much additional work is required to account for every user identity on a server.
- Filters simplify ongoing identity management by listing only desired user identities. Both user identities that will not be mapped and user identities that have already been mapped can be hidden. If both types of these user identities are hidden then only those user identities that still require action are listed.
- Quickly subset and locate people and registries in long lists using filters.

Population

As mentioned above, TIM SSO provides simple and flexible wizards to import data and populate an EIM domain for EIM identifiers or people, registries or servers, and user identities. This is accomplished by importing data in comma separated value (.csv) files. Also, user identities can be loaded directly from properly configured LDAP servers.

Starting with a .csv file like the one displayed in Figure E-15, People .csv import file, TIM SSO imports the listed people and populates the current EIM domain.



	A	B	C	D	E	F	G	I
1	Import	People	Description	FirstName	MiddleName	LastName	Alias	Employee Number
7	Yes	Emily S. Sanders	President and Chief Executive Offi	Emily	Savannah	Sanders	Emma Sanders	L1948B
8	Yes	Adrian O. Turner	SVP and General Counsel	Adrian	Oscar	Turner		A1439X
9	Yes	Xavier A. Peterson	Director, Import and Export Comp	Xavier	Aaron	Peterson		L1807O
10	Yes	Bryan J. Long	Legal Assistant	Bryan	Joshua	Long		E1732S
11	Yes	Maria B. Anderson	Legal Assistant	Maria	Bailey	Anderson		U1208D
12	Yes	Jeremiah J. Roberts	SVP, Chief Operating Officer (CO(	Jeremiah	Jason	Roberts	Jerry Roberts	U1198X
13	Yes	Jennifer E. Hayes	Director, Information Technology	Jennifer	Elizabeth	Hayes	Jenny Hayes	E1360W
14	Yes	Brian G. Baker	Manager, Servers	Brian	Garrett	Baker		N1821A
15	Yes	Juan P. Martin	Senior Systems Analyst	Juan	Patrick	Martin		A1406Q
16	Yes	Abigail C. Hall	Performance and Capacity Analys	Abigail	Christina	Hall	Abby Hall	U1386Z

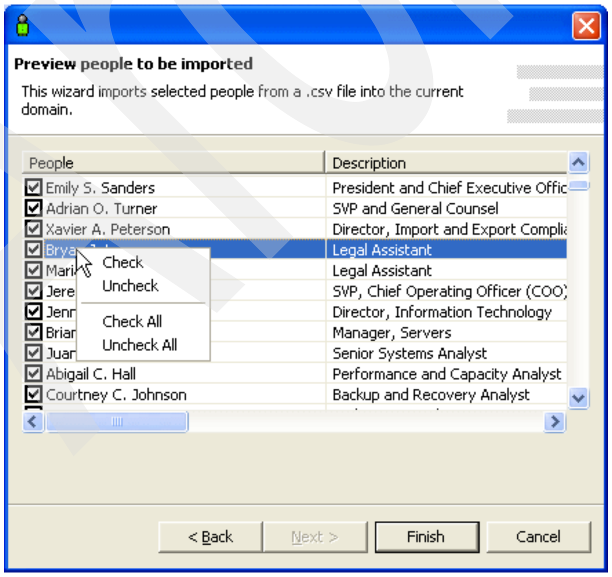
Figure E-15 People .csv import file

In this example the following information will be imported and used to populate the EIM domain:

- People (EIM identifier)
- People description
- Additional information for each person:
  - First name
  - Middle name
  - Last name
  - Employee number
- Any aliases

Additional information is required for the collation to follow. The additional information is used to generate suggestions, which are user identities across the enterprise that are potential associations for people. Suggestions are based on a set of customizable user identity composition rules. It is important, therefore, to understand how user identities are constructed on all registries in the EIM domain prior to importing people.

The People import wizard provides a preview step to ensure accurate population of the EIM domain. This step is shown in Figure E-16, Preview people to be imported.



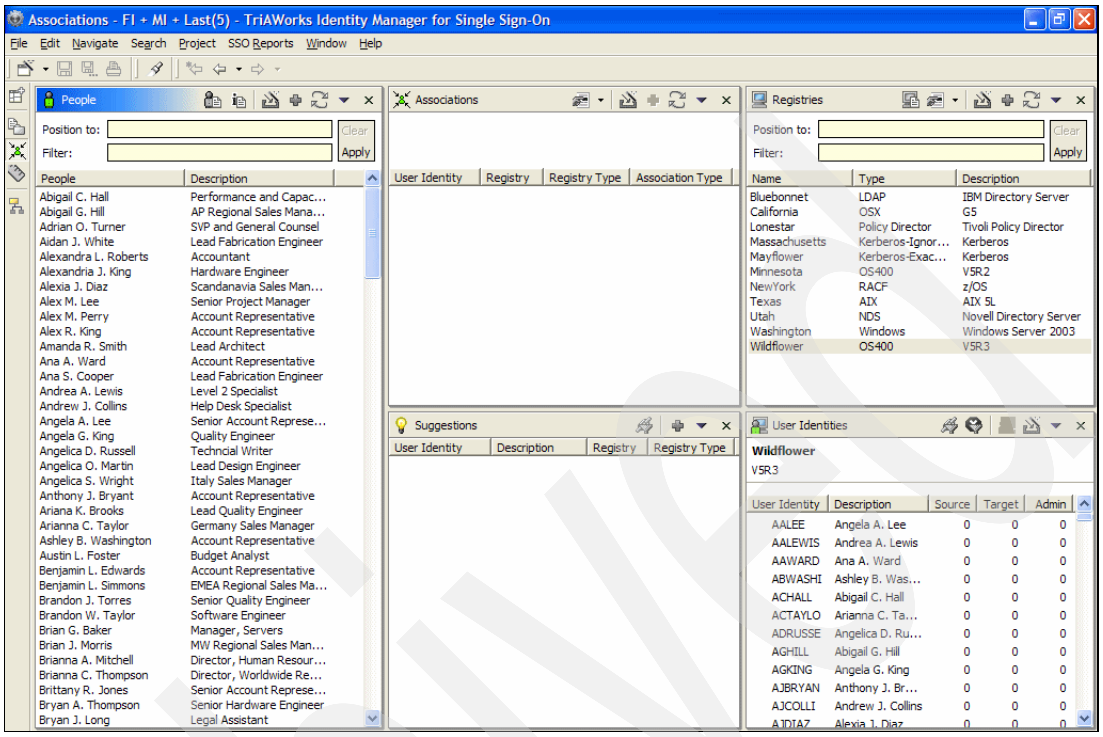
Preview people to be imported

This wizard imports selected people from a .csv file into the current domain.

People	Description
☑ Emily S. Sanders	President and Chief Executive Offic
☑ Adrian O. Turner	SVP and General Counsel
☑ Xavier A. Peterson	Director, Import and Export Complia
☑ Brya	Legal Assistant
☑ Mari	Legal Assistant
☑ Jere	SVP, Chief Operating Officer (COO)
☑ Jenr	Director, Information Technology
☑ Briar	Manager, Servers
☑ Juar	Senior Systems Analyst
☑ Abigail C. Hall	Performance and Capacity Analyst
☑ Courtney C. Johnson	Backup and Recovery Analyst

Figure E-16 Preview people to be imported

After importing registries, people, and user identities for each registry, TIM SSO will look like Figure E-17, Associations perspective after populating.



People	Description
Abigail C. Hall	Performance and Capac…
Abigail G. Hill	AP Regional Sales Mana…
Adrian O. Turner	SVP and General Counsel
Aidan J. White	Lead Fabrication Engineer
Alexandra L. Roberts	Accountant
Alexandria J. King	Hardware Engineer
Alexia J. Diaz	Scandanavia Sales Man…
Alex M. Lee	Senior Project Manager
Alex M. Perry	Account Representative
Alex R. King	Account Representative
Amanda R. Smith	Lead Architect
Ana A. Ward	Account Representative
Ana S. Cooper	Lead Fabrication Engineer
Andrea A. Lewis	Level 2 Specialist
Andrew J. Collins	Help Desk Specialist
Angela A. Lee	Senior Account Represe…
Angela G. King	Quality Engineer
Angelica D. Russell	Techncial Writer
Angelica O. Martin	Lead Design Engineer
Angelica S. Wright	Italy Sales Manager
Anthony J. Bryant	Account Representative
Ariana K. Brooks	Lead Quality Engineer
Arianna C. Taylor	Germany Sales Manager
Ashley B. Washington	Account Representative
Austin L. Foster	Budget Analyst
Benjamin L. Edwards	Account Representative
Benjamin L. Simmons	EMEA Regional Sales Ma…
Brandon J. Torres	Senior Quality Engineer
Brandon W. Taylor	Software Engineer
Brian G. Baker	Manager, Servers
Brian J. Morris	MW Regional Sales Man…
Brianna A. Mitchell	Director, Human Resour…
Brianna C. Thompson	Director, Worldwide Re…
Brittany R. Jones	Senior Account Represe…
Bryan A. Thompson	Senior Hardware Engineer
Bryan J. Long	Legal Assistant

User Identity	Registry	Registry Type	Association Type

Name	Type	Description
Bluebonnet	LDAP	IBM Directory Server
California	OSX	G5
Lonestar	Policy Director	Tivoli Policy Director
Massachusetts	Kerberos-Ignor…	Kerberos
Mayflower	Kerberos-Exac…	Kerberos
Minnesota	OS400	V5R2
NewYork	RACF	z/OS
Texas	AIX	AIX 5L
Utah	NDS	Novell Directory Server
Washington	Windows	Windows Server 2003
Wildflower	OS400	V5R3

User Identity	Description	Registry	Registry Type

Wildflower
V5R3

User Identity	Description	Source	Target	Admin
AALEE	Angela A. Lee	0	0	0
AALEWIS	Andrea A. Lewis	0	0	0
AAWARD	Ana A. Ward	0	0	0
ABWASHI	Ashley B. Was…	0	0	0
ACHALL	Abigail C. Hall	0	0	0
ACTAYLO	Arianna C. Ta…	0	0	0
ADRUSSE	Angelica D. Ru…	0	0	0
AGHILL	Abigail G. Hill	0	0	0
AGKING	Angela G. King	0	0	0
AJBRYAN	Anthony J. Br…	0	0	0
AJCOLLI	Andrew J. Collins	0	0	0
AJDIAZ	Alexia J. Diaz	0	0	0

Figure E-17 Associations perspective after populating

There are five views displayed in the Associations perspective (which is simply a collection of views):

- The *People* view manages the list of people (EIM identifiers) for the selected EIM domain.
- The *Associations* view manages the list of associations (mappings) for the selected person.
- The *Registries* view manages the list of registries (servers) for the selected EIM domain.
- The *User Identities* view manages the list of user identities (users) for the selected registry.
- The *Suggestions* view lists user identities that are potential associations based on a set of customizable user identity composition rules.

The People, Registries, and User Identities views contain information after populating the EIM domain. The Associations and Suggestions views will contain information when mapping or collating user identities and people begins.

Collation

Mapping user identities to people is done one of two ways in TIM SSO:

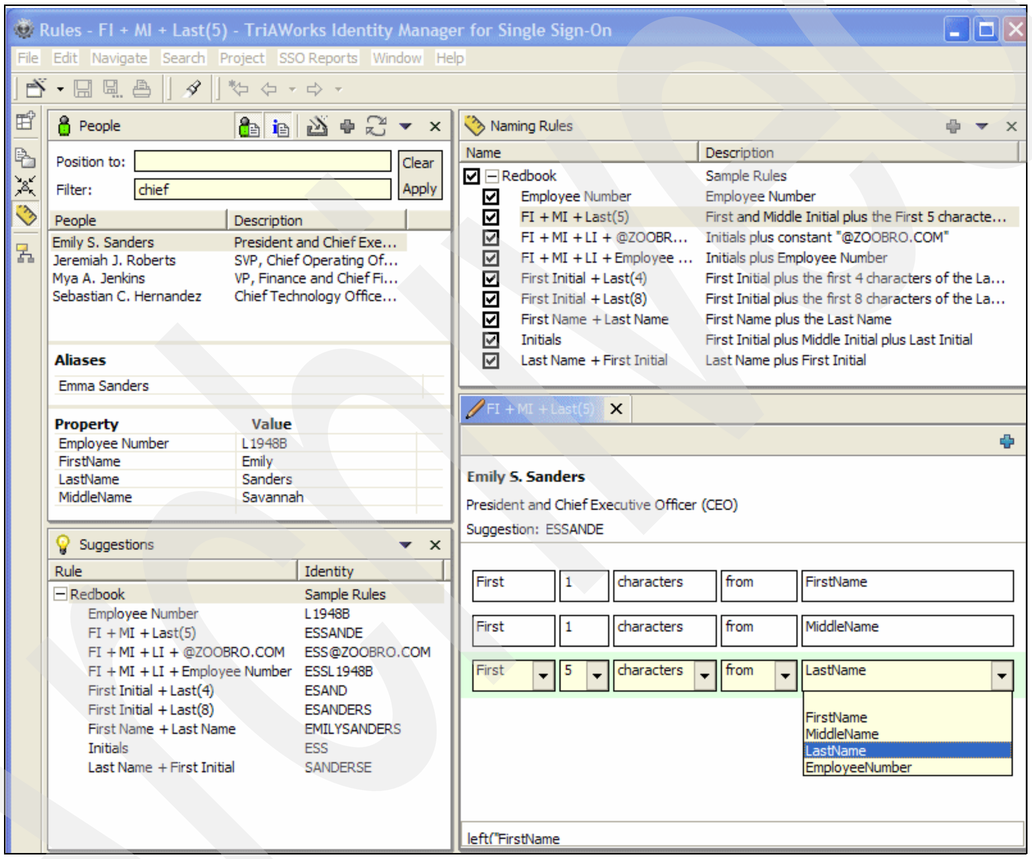
- Manually, using drag-and-drop, or using the Create New Association wizard.
- Automatically, using the Import Associations wizard in conjunction with the customizable user identity composition rules.

Without the ability to automatically create associations the burden of managing an EIM domain is daunting. For this reason it is important to understand how user identities are constructed across the enterprise. With this understanding, the additional information for people can be organized so rules are created to generate accurate suggestions.

Rules

As shown in Figure E-18, Rules perspective, the Rules perspective is comprised of four views:

- The People view is the same as the one found in the Associations perspective.
- The Naming Rules view manages the rules and rule sets used to generate suggestions.
- The Individual Rule view is where rules are edited.
- The Suggestions view lists the user identities created as potential associations for the selected person given the selected rules in the Naming Rules view.



Rules - FI + MI + Last(5) - TriAWorks Identity Manager for Single Sign-On
File Edit Navigate Search Project SSO Reports Window Help
People
Clear
Apply
Naming Rules
Suggestions
FI + MI + Last(5)

Position to:
Filter: chief

People	Description
Emily S. Sanders	President and Chief Exe…
Jeremiah J. Roberts	SVP, Chief Operating Of…
Mya A. Jenkins	VP, Finance and Chief Fi…
Sebastian C. Hernandez	Chief Technology Office…

Aliases
Emma Sanders

Property	Value
Employee Number	L1948B
FirstName	Emily
LastName	Sanders
MiddleName	Savannah

Rule	Identity
Redbook	Sample Rules
Employee Number	L1948B
FI + MI + Last(5)	ESSANDE
FI + MI + LI + @ZOOBRO.COM	ESS@ZOOBRO.COM
FI + MI + LI + Employee Number	ESSL1948B
First Initial + Last(4)	ESAND
First Initial + Last(8)	ESANDERS
First Name + Last Name	EMILYSANDERS
Initials	ESS
Last Name + First Initial	SANDERSE

Name	Description
Redbook	Sample Rules
Employee Number	Employee Number
FI + MI + Last(5)	First and Middle Initial plus the First 5 characte…
FI + MI + LI + @ZOOBR…	Initials plus constant "@ZOOBRO.COM"
FI + MI + LI + Employee …	Initials plus Employee Number
First Initial + Last(4)	First Initial plus the first 4 characters of the La…
First Initial + Last(8)	First Initial plus the first 8 characters of the La…
First Name + Last Name	First Name plus the Last Name
Initials	First Initial plus Middle Initial plus Last Initial
Last Name + First Initial	Last Name plus First Initial

Emily S. Sanders
President and Chief Executive Officer (CEO)
Suggestion: ESSANDE

First	1	characters	from	FirstName
First	1	characters	from	MiddleName
First	5	characters	from	LastName

FirstName
MiddleName
LastName
EmployeeNumber

left("FirstName

Figure E-18 Rules perspective

In this example nine different rules have been created based on the additional information supplied with each person.

The "FI + MI + Last(5)" rule is selected and displayed for editing in the individual rule view. This rule constructs a user identity comprised of a person's First Initial, followed by their Middle Initial, and then followed by the first 5 characters of their last name.

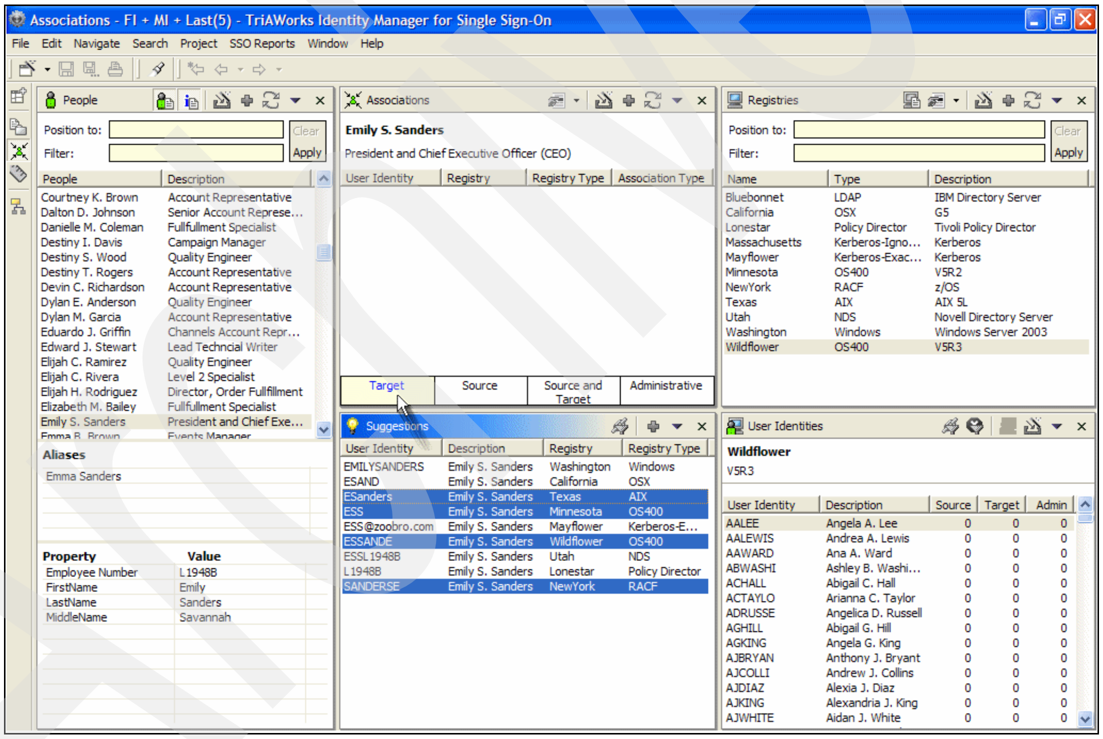
Since the person Emily S. Sanders is selected, the rule editor displays her suggested user identity as the "FI + MI + Last(5)" rule is being constructed. The suggestion for Emily S. Sanders is ESSANDE. Likewise, the Suggestions view lists all the user identities generated for, in this case, Emily S. Sanders and the currently selected rules. The Suggestions view is powerful in that it enables rules to be verified prior to using them to make associations.

Also, the filter function is used in this example. There may be many people in a company, so filtering is used to subset the people list and quickly locate someone. Since Emily S. Sanders is the CEO of the organization, a filter is used to find all people with “chief” in their name, description, alias, or additional information. Emily S. Sanders, it turns out, is one of four people that meets this criteria.

Manually Creating Associations

It is important to note up-front the relationships between the different views in the Associations perspective. When a new person is selected in the People view, the Associations and Suggestions views are also updated to show the respective information for that person in that view. Likewise, when a new registry is selected in the Registry view the User Identities view is updated to list the user identities for that registry.

Associations can always be created manually at any time by selecting one or more user identities in the Suggestions or User Identities view and then invoking the Create New Association wizard. This is accomplished either by right-clicking on one of the selected user identities or by performing a drag-and drop with them to the selected person in the Associations view. The drag-and-drop method is shown in Figure E-19, Drag-and-drop associations.


Associations - FI + MI + Last(5) - TriAWorks Identity Manager for Single Sign-On
File Edit Navigate Search Project SSO Reports Window Help
People
Position to:
Filter:
Clear
Apply
Associations
Registries
Target
Source
Source and Target
Administrative
Suggestions
User Identities

People	Description
Courtney K. Brown	Account Representative
Dalton D. Johnson	Senior Account Represe...
Danielle M. Coleman	Fulfullment Specialist
Destiny I. Davis	Campaign Manager
Destiny S. Wood	Quality Engineer
Destiny T. Rogers	Account Representative
Devin C. Richardson	Account Representative
Dylan E. Anderson	Quality Engineer
Dylan M. Garcia	Account Representative
Eduardo J. Griffin	Channels Account Repr...
Edward J. Stewart	Lead Techncial Writer
Elijah C. Ramirez	Quality Engineer
Elijah C. Rivera	Level 2 Specialist
Elijah H. Rodriguez	Director, Order Fullfillment
Elizabeth M. Bailey	Fullfullment Specialist
Emily S. Sanders	President and Chief Exe...
Emma B. Brown	Events Manager

Aliases
Emma Sanders

Property	Value
Employee Number	L1948B
FirstName	Emily
LastName	Sanders
MiddleName	Savannah

Emily S. Sanders
President and Chief Executive Officer (CEO)

User Identity	Registry	Registry Type	Association Type

User Identity	Description	Registry	Registry Type
EMILYSANDERS	Emily S. Sanders	Washington	Windows
ESAND	Emily S. Sanders	California	OSX
ESanders	Emily S. Sanders	Texas	AIX
ESS	Emily S. Sanders	Minnesota	OS400
ESS@zoobro.com	Emily S. Sanders	Mayflower	Kerberos-E...
ESSANDE	Emily S. Sanders	Wildflower	OS400
ESSL1948B	Emily S. Sanders	Utah	NDS
L1948B	Emily S. Sanders	Lonestar	Policy Director
SANDERSE	Emily S. Sanders	NewYork	RACF

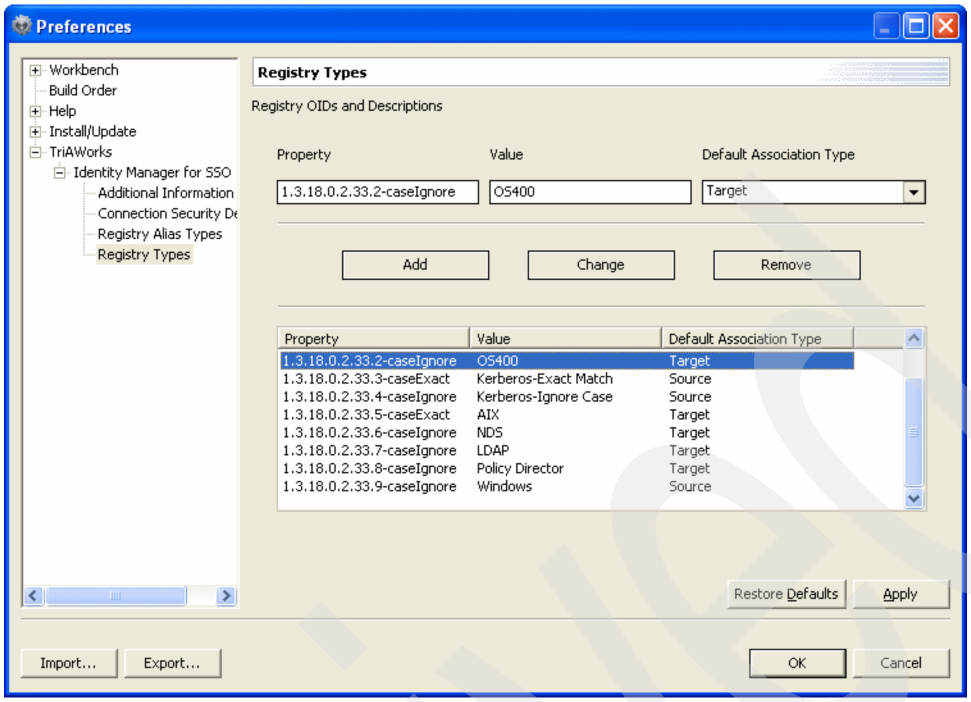
Name	Type	Description
Bluebonnet	LDAP	IBM Directory Server
California	OSX	G5
Lonestar	Policy Director	Tivoli Policy Director
Massachusetts	Kerberos-Igno...	Kerberos
Mayflower	Kerberos-Exac...	Kerberos
Minnesota	OS400	V5R2
NewYork	RACF	z/OS
Texas	AIX	AIX 5L
Utah	NDS	Novell Directory Server
Washington	Windows	Windows Server 2003
Wildflower	OS400	V5R3

Wildflower
V5R3

User Identity	Description	Source	Target	Admin
AALEE	Angela A. Lee	0	0	0
AALEWIS	Andrea A. Lewis	0	0	0
AAWARD	Ana A. Ward	0	0	0
ABWASHI	Ashley B. Washi...	0	0	0
ACHALL	Abigail C. Hall	0	0	0
ACTAYLO	Arianna C. Taylor	0	0	0
ADRUSSE	Angelica D. Russell	0	0	0
AGHILL	Abigail G. Hill	0	0	0
AGKING	Angela G. King	0	0	0
AJBRYAN	Anthony J. Bryant	0	0	0
AJCOLLI	Andrew J. Collins	0	0	0
AJDIAZ	Alexia J. Diaz	0	0	0
AJKING	Alexandria J. King	0	0	0
AJWHITE	Aidan J. White	0	0	0

Figure E-19 Drag-and-Drop Associations

In this example, the selected suggestions are dragged to the target “drop zone,” meaning target associations will be created between all these suggestions and the person Emily S. Sanders. If, on the other hand, the suggestions are dropped outside a “drop zone” within the Associations view then the association types will be set to the registry default. The Registry default association types are configurable, per Figure E-20, Registry Types. In this example, by default, source associations would be created with user identities from Windows servers, target associations would be created with user identities from an iSeries, and so on.



Registry Types

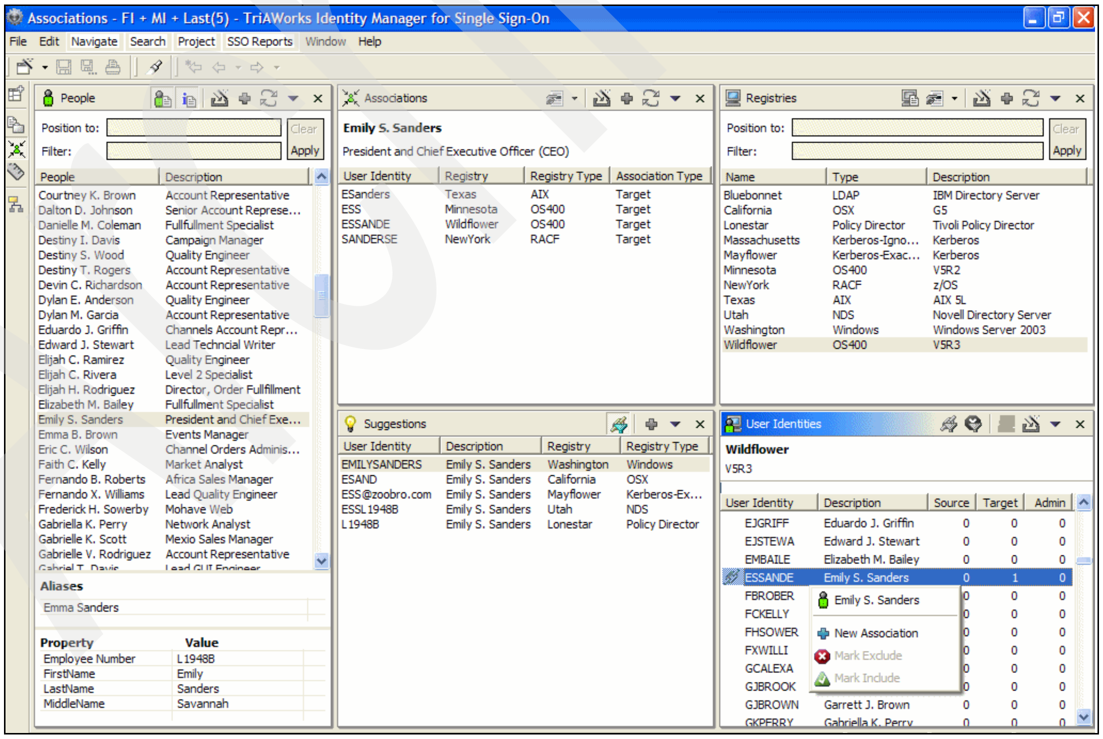
Registry OIDs and Descriptions

Property: 1.3.18.0.2.33.2-caseIgnore
Value: OS400
Default Association Type: Target

Property	Value	Default Association Type
1.3.18.0.2.33.2-caseIgnore	OS400	Target
1.3.18.0.2.33.3-caseExact	Kerberos-Exact Match	Source
1.3.18.0.2.33.4-caseIgnore	Kerberos-Ignore Case	Source
1.3.18.0.2.33.5-caseExact	AIX	Target
1.3.18.0.2.33.6-caseIgnore	NDS	Target
1.3.18.0.2.33.7-caseIgnore	LDAP	Target
1.3.18.0.2.33.8-caseIgnore	Policy Director	Target
1.3.18.0.2.33.9-caseIgnore	Windows	Source

Figure E-20 Registry types

After manually creating the example target associations using drag-and-drop TIM SSO looks like Figure E-21, After creating drag-and-drop associations.



People	Description
Courtney K. Brown	Account Representative
Dalton D. Johnson	Senior Account Represe...
Danielle M. Coleman	Fullfullment Specialist
Destiny I. Davis	Campaign Manager
Destiny S. Wood	Quality Engineer
Destiny T. Rogers	Account Representative
Devin C. Richardson	Account Representative
Dylan E. Anderson	Quality Engineer
Dylan M. Garcia	Account Representative
Eduardo J. Griffin	Channels Account Repr...
Edward J. Stewart	Lead Techncial Writer
Elijah C. Ramirez	Quality Engineer
Elijah C. Rivera	Level 2 Specialist
Elijah H. Rodriguez	Director, Order Fullfillment
Elizabeth M. Bailey	Fullfullment Specialist
Emily S. Sanders	President and Chief Exe...
Emma B. Brown	Events Manager
Eric C. Wilson	Channel Orders Adminis...
Faith C. Kelly	Market Analyst
Fernando B. Roberts	Africa Sales Manager
Fernando X. Williams	Lead Quality Engineer
Frederick H. Sowerby	Mohave Web
Gabriella K. Perry	Network Analyst
Gabrielle K. Scott	Mexio Sales Manager
Gabrielle V. Rodriguez	Account Representative
Gabriel T. Davis	Lead GUI Engineer

Aliases

Emma Sanders

Property	**Value**
Employee Number	L1948B
FirstName	Emily
LastName	Sanders
MiddleName	Savannah

Emily S. Sanders

President and Chief Executive Officer (CEO)

User Identity	Registry	Registry Type	Association Type
ESanders	Texas	AIX	Target
ESS	Minnesota	OS400	Target
ESSANDE	Wildflower	OS400	Target
SANDERSE	NewYork	RACF	Target

User Identity	Description	Registry	Registry Type
EMILYSANDERS	Emily S. Sanders	Washington	Windows
ESAND	Emily S. Sanders	California	OSX
ESS@zoobro.com	Emily S. Sanders	Mayflower	Kerberos-Ex...
ESSL1948B	Emily S. Sanders	Utah	NDS
L1948B	Emily S. Sanders	Lonestar	Policy Director

Name	Type	Description
Bluebonnet	LDAP	IBM Directory Server
California	OSX	G5
Lonestar	Policy Director	Tivoli Policy Director
Massachusetts	Kerberos-Igno...	Kerberos
Mayflower	Kerberos-Exac...	Kerberos
Minnesota	OS400	V5R2
NewYork	RACF	z/OS
Texas	AIX	AIX 5L
Utah	NDS	Novell Directory Server
Washington	Windows	Windows Server 2003
Wildflower	OS400	V5R3

Wildflower

V5R3

User Identity	Description	Source	Target	Admin
EJGRIFF	Eduardo J. Griffin	0	0	0
EJSTEWA	Edward J. Stewart	0	0	0
EMBAILE	Elizabeth M. Bailey	0	0	0
ESSANDE	Emily S. Sanders	0	1	0
FBROBER		0	0	0
FCKELLY		0	0	0
FHSOWER		0	0	0
FXWILLI		0	0	0
GCALEXA		0	0	0
GJBROOK		0	0	0
GJBROWN	Garrett J. Brown	0	0	0
GKPERRY	Gabriella K. Perry	0	0	0

Figure E-21 After creating drag-and-drop associations

There are now three notable features displayed:

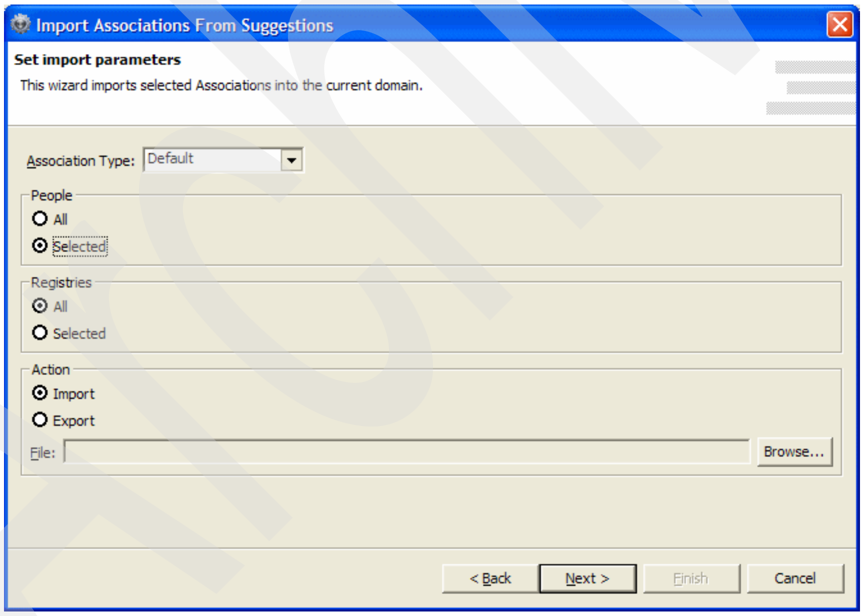
- With filtering turned on, the Suggestions view only lists those suggestions not yet mapped to a person.
- The ESSANDE user identity on the Wildflower registry is marked with a chain link to denote it is mapped to a person. In fact, the association type counts show it is mapped with 1 target association.
- Right-clicking on the ESSANDE user identity shows it is mapped to the Emily S. Sanders person.

Automatically Creating Association

Rules help make manual associations faster and more accurate by reducing the time to find the user identities that might potentially map to a person.

Likewise, rules enable numerous associations to be created automatically in a series of steps that take only a few minutes. In fact, if desired, it is possible to automatically map all people and user identities across the enterprise at once. Of course, the ability to do so depends on the consistency of user identities on registries and how soundly rules are created.

Given sufficient up-front planning time to define rules that match how user identities are constructed across the enterprise, collating and populating an EIM domain is very easy with TIM SSO. All that is required is to run the Import Associations wizard and select the "Import from Suggestions" option. Figure E-22 shows the Import Associations wizard run using a selected subset of people and all registries.


Import Associations From Suggestions

Set import parameters

This wizard imports selected Associations into the current domain.

Association Type: Default

People: All [✓ Selected]

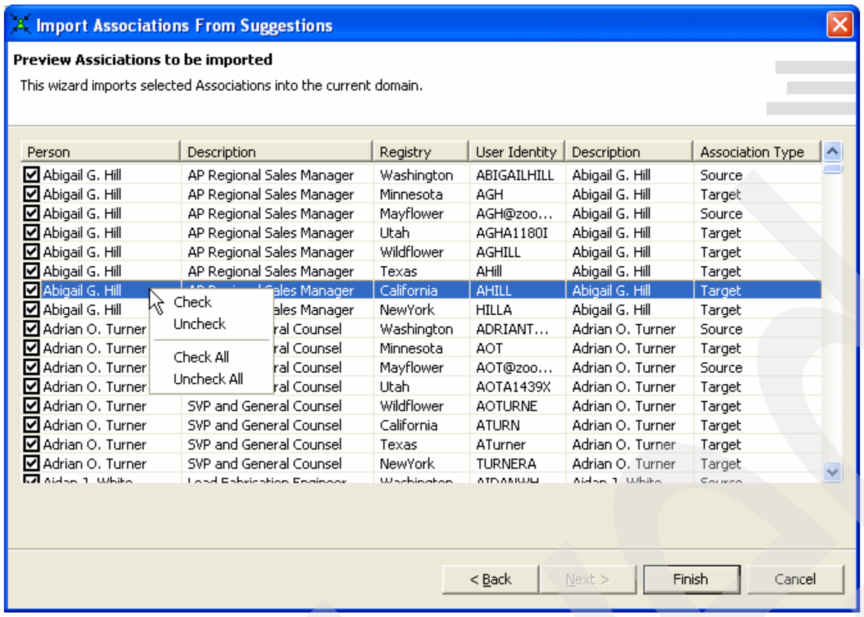
Registries: [✓ All] Selected

Action: [✓ Import] Export

File:

Figure E-22 Set import parameters

This means for the subset of people selected, TIM SSO will populate the EIM domain with the suggested associations for all user identities across the enterprise. This is a powerful function, so per Figure E-23, Preview associations to be imported, a preview step is next provided by the Import Associations wizard. This step provides an opportunity to deselect any suggested associations before they are populated to the EIM domain.



Preview Assiciations to be imported

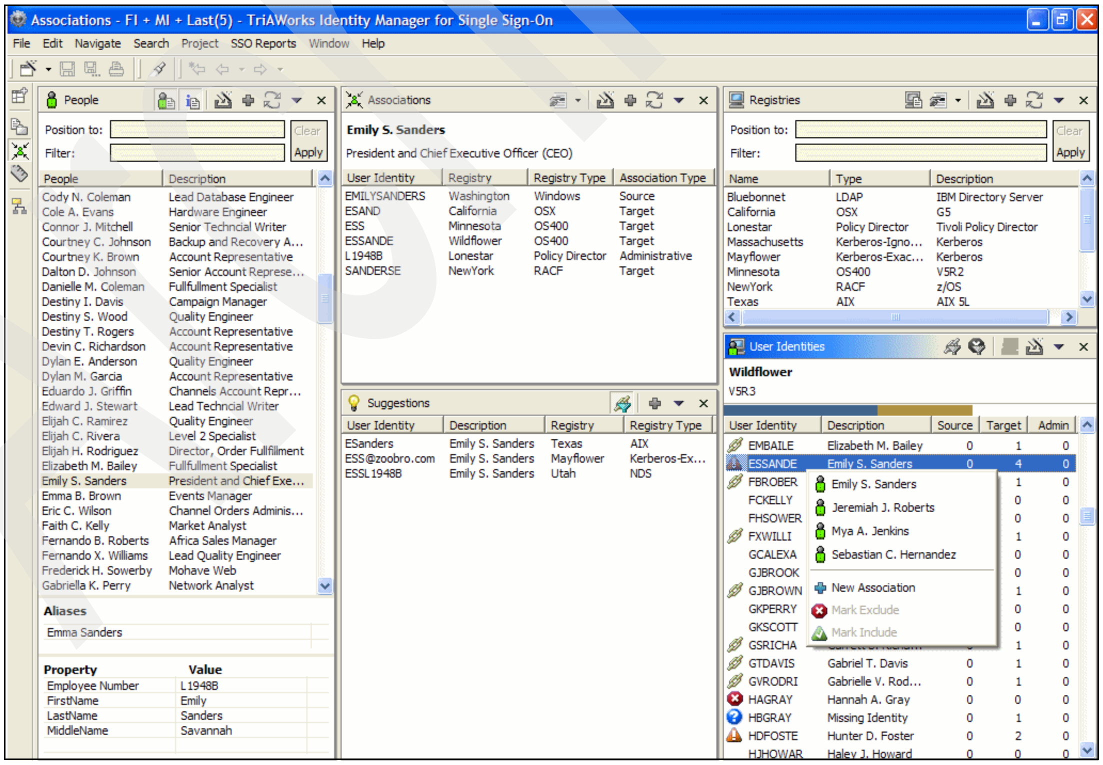
This wizard imports selected Associations into the current domain.

Person	Description	Registry	User Identity	Description	Association Type
☑ Abigail G. Hill	AP Regional Sales Manager	Washington	ABIGAILHILL	Abigail G. Hill	Source
☑ Abigail G. Hill	AP Regional Sales Manager	Minnesota	AGH	Abigail G. Hill	Target
☑ Abigail G. Hill	AP Regional Sales Manager	Mayflower	AGH@zoo…	Abigail G. Hill	Source
☑ Abigail G. Hill	AP Regional Sales Manager	Utah	AGHA1180I	Abigail G. Hill	Target
☑ Abigail G. Hill	AP Regional Sales Manager	Wildflower	AGHILL	Abigail G. Hill	Target
☑ Abigail G. Hill	AP Regional Sales Manager	Texas	AHill	Abigail G. Hill	Target
☑ Abigail G. Hill	ales Manager	California	AHILL	Abigail G. Hill	Target
☑ Abigail G. Hill	ales Manager	NewYork	HILLA	Abigail G. Hill	Target
☑ Adrian O. Turner	ral Counsel	Washington	ADRIANT…	Adrian O. Turner	Source
☑ Adrian O. Turner	ral Counsel	Minnesota	AOT	Adrian O. Turner	Target
☑ Adrian O. Turner	ral Counsel	Mayflower	AOT@zoo…	Adrian O. Turner	Source
☑ Adrian O. Turner	ral Counsel	Utah	AOTA1439X	Adrian O. Turner	Target
☑ Adrian O. Turner	SVP and General Counsel	Wildflower	AOTURNE	Adrian O. Turner	Target
☑ Adrian O. Turner	SVP and General Counsel	California	ATURN	Adrian O. Turner	Target
☑ Adrian O. Turner	SVP and General Counsel	Texas	ATurner	Adrian O. Turner	Target
☑ Adrian O. Turner	SVP and General Counsel	NewYork	TURNERA	Adrian O. Turner	Target

Figure E-23   Preview Associations to be imported

Also, rest assured that even after associations are made, whether manually or automatically, they can also be easily modified or deleted with TIM SSO.

After automatically creating the automated associations the EIM domain is represented by Figure E-24, Automated associations created.



People	Description
Cody N. Coleman	Lead Database Engineer
Cole A. Evans	Hardware Engineer
Connor J. Mitchell	Senior Techncial Writer
Courtney C. Johnson	Backup and Recovery A…
Courtney K. Brown	Account Representative
Dalton D. Johnson	Senior Account Represe…
Danielle M. Coleman	Fullfullment Specialist
Destiny I. Davis	Campaign Manager
Destiny S. Wood	Quality Engineer
Destiny T. Rogers	Account Representative
Devin C. Richardson	Account Representative
Dylan E. Anderson	Quality Engineer
Dylan M. Garcia	Account Representative
Eduardo J. Griffin	Channels Account Repr…
Edward J. Stewart	Lead Techncial Writer
Elijah C. Ramirez	Quality Engineer
Elijah C. Rivera	Level 2 Specialist
Elijah H. Rodriguez	Director, Order Fullfillment
Elizabeth M. Bailey	Fullfullment Specialist
Emily S. Sanders	President and Chief Exe…
Emma B. Brown	Events Manager
Eric C. Wilson	Channel Orders Adminis…
Faith C. Kelly	Market Analyst
Fernando B. Roberts	Africa Sales Manager
Fernando X. Williams	Lead Quality Engineer
Frederick H. Sowerby	Mohave Web
Gabriella K. Perry	Network Analyst

Aliases

Emma Sanders

Property	Value
Employee Number	L1948B
FirstName	Emily
LastName	Sanders
MiddleName	Savannah

Emily S. Sanders

President and Chief Executive Officer (CEO)

User Identity	Registry	Registry Type	Association Type
EMILYSANDERS	Washington	Windows	Source
ESAND	California	OSX	Target
ESS	Minnesota	OS400	Target
ESSANDE	Wildflower	OS400	Target
L1948B	Lonestar	Policy Director	Administrative
SANDERSE	NewYork	RACF	Target

User Identity	Description	Registry	Registry Type
ESanders	Emily S. Sanders	Texas	AIX
ESS@zoobro.com	Emily S. Sanders	Mayflower	Kerberos-Ex…
ESSL1948B	Emily S. Sanders	Utah	NDS

Name	Type	Description
Bluebonnet	LDAP	IBM Directory Server
California	OSX	G5
Lonestar	Policy Director	Tivoli Policy Director
Massachusetts	Kerberos-Igno…	Kerberos
Mayflower	Kerberos-Exac…	Kerberos
Minnesota	OS400	V5R2
NewYork	RACF	z/OS
Texas	AIX	AIX 5L

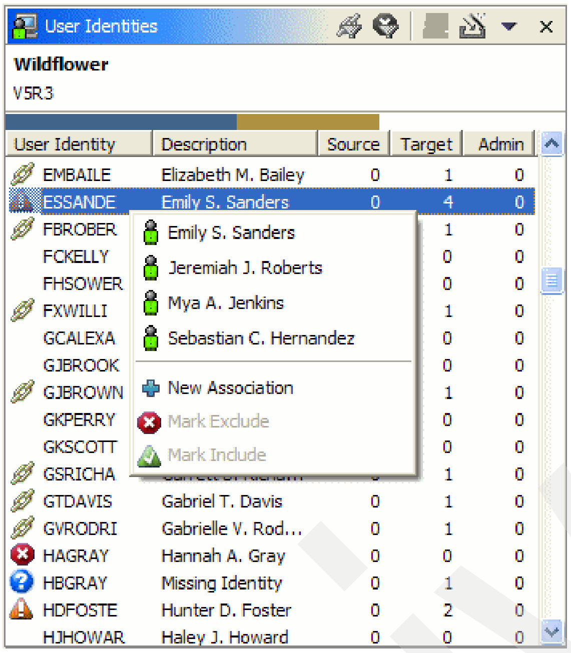
Wildflower

V5R3

User Identity	Description	Source	Target	Admin
EMBAILE	Elizabeth M. Bailey	0	1	0
ESSANDE	Emily S. Sanders	0	4	0
FBROBER	Emily S. Sanders		1	0
FCKELLY	Jeremiah J. Roberts		0	0
FHSOWER	Mya A. Jenkins		0	0
FXWILLI	Sebastian C. Hernandez		1	0
GCALEXA			0	0
GJBROOK			0	0
GJBROWN			1	0
GKPERRY			0	0
GKSCOTT			0	0
GSRICHA			1	0
GTDAVIS	Gabriel T. Davis	0	1	0
GVRODRI	Gabrielle V. Rod…	0	1	0
HAGRAY	Hannah A. Gray	0	0	0
HBGRAY	Missing Identity	0	1	0
HDFOSTE	Hunter D. Foster	0	2	0
HJHOWAR	Haley J. Howard	0	0	0

Figure E-24   Automated associations created

The User Identities view, enlarged in Figure E-25, User Identity view, now shows several user identities marked with a chain link to denote they are mapped to a person.


User Identities

Wildflower

V5R3

User Identity	Description	Source	Target	Admin
EMBAILE	Elizabeth M. Bailey	0	1	0
ESSANDE	Emily S. Sanders	0	4	0
FBROBER			1	0
FCKELLY			0	0
FHSOWER			0	0
FXWILLI			1	0
GCALEXA			0	0
GJBROOK			0	0
GJBROWN			1	0
GKPERRY			0	0
GKSCOTT			0	0
GSRICHA	[illegible]		1	0
GTDAVIS	Gabriel T. Davis	0	1	0
GVRODRI	Gabrielle V. Rod...	0	1	0
HAGRAY	Hannah A. Gray	0	0	0
HBGRAY	Missing Identity	0	1	0
HDFOSTE	Hunter D. Foster	0	2	0
HJHOWAR	Haley J. Howard	0	0	0

Figure E-25   User identity view

The User Identity view now also reveals several other features to help simplify the management of an EIM domain:

- The orange triangle '!' icon next to the HDFOSTE and ESSANDE user identities warns these user identities are mapped to more than one person. While EIM allows this, it is not a best-practice. Right-clicking on the ESSANDE user identity reveals the four (4) people to whom she is mapped: Emily S. Sanders (the correct person), Jeremiah J. Roberts, Mya A. Jenkins, and Sebastian C. Hernandez. Clicking on any one of these names will select that person in the People, Associations, and Suggestions views so corrective action can then be taken.
- The blue circle '?' icon next to the HBGRAY user identity (also, its description reads, "Missing Identity") warns this user identity is currently mapped to a person even though it does not actually exist on the registry. Right-clicking on the user identity reveals the person the user identity is mapped to, which can then be selected so corrective action can be taken.
- The red circle 'X' icon next to the HAGRAY user identity denotes this user identity has been marked excluded. An administrator is able to mark user identities as excluded to indicate they will not be mapped in the foreseeable future. To simplify managing large numbers of user identities, this view provides the ability to filter whether currently associated and/or excluded identities are displayed. Consequently, it is possible for this view to only list the user identities that likely require some action – the user identities that are not marked excluded and not yet mapped.
- The blue and brown horizontal stacked bar "dashboard" graph near the top of the User Identities view provides a quick indicator of how many user identities are and are not mapped on the selected registry.
  - The blue portion of the bar represents the percentage of user identities on the selected registry that are currently mapped to a person.

- – The brown portion of the bar represents the percentage of identities on the selected registry that are currently marked excluded.
- – The white portion of the bar, therefore, represents the percentage of identities on the selected registry that may need to be addressed, as they are neither mapped nor excluded.

In this example, on the Wildflower registry, approximately 40% of the user identities are mapped, 30% of the user identities are marked excluded (including all the IBM supplied ‘Q’ profiles), and the remaining 30% of the user identities may require action.

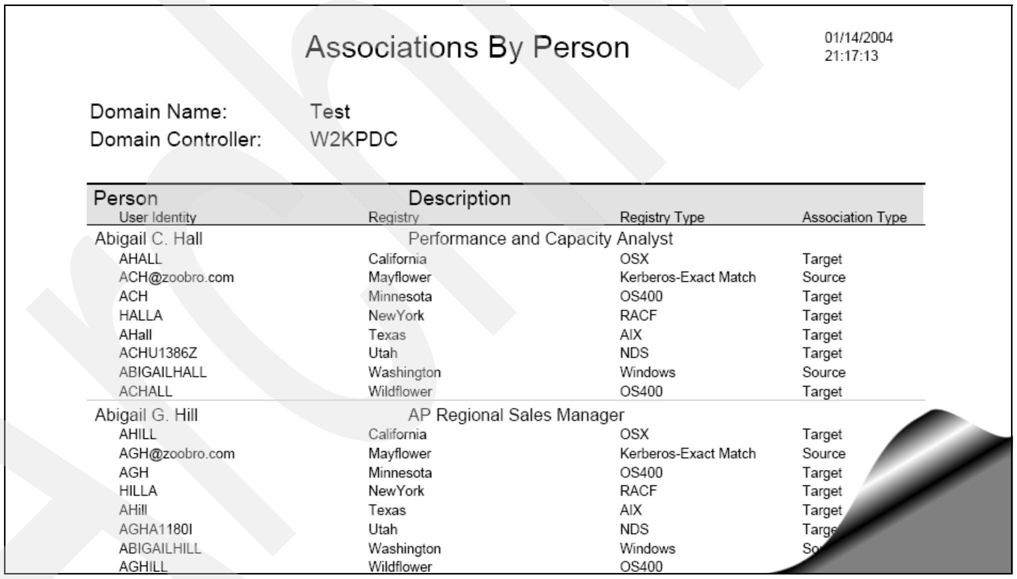
Reports

Numerous Adobe® .PDF reports may be generated to record and/or share the status of the EIM domain.

Reports can be run in one of two ways:

- ► When reports are run from the TIM SSO main menu then all people, user identities, or registries are selected for the report.
- ► When reports are run as right-clicks from their respective individual views then the only the selected people, user identities, or registries are included in the report.

As an example, the report below (Figure E-26 and Figure E-27) lists all the people in the EIM domain and their associations.



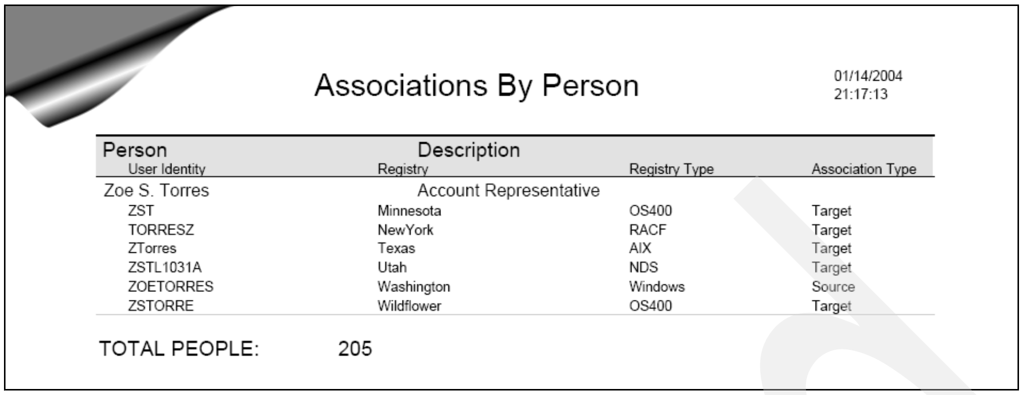
Associations By Person

01/14/2004
21:17:13

Domain Name: Test
Domain Controller: W2KPDC

Person	Description		
User Identity	Registry	Registry Type	Association Type
Abigail C. Hall	Performance and Capacity Analyst		
AHALL	California	OSX	Target
ACH@zoobro.com	Mayflower	Kerberos-Exact Match	Source
ACH	Minnesota	OS400	Target
HALLA	NewYork	RACF	Target
AHall	Texas	AIX	Target
ACHU1386Z	Utah	NDS	Target
ABIGAILHALL	Washington	Windows	Source
ACHALL	Wildflower	OS400	Target
Abigail G. Hill	AP Regional Sales Manager		
AHILL	California	OSX	Target
AGH@zoobro.com	Mayflower	Kerberos-Exact Match	Source
AGH	Minnesota	OS400	Target
HILLA	NewYork	RACF	Target
AHill	Texas	AIX	Target
AGHA1180I	Utah	NDS	Targe
ABIGAILHILL	Washington	Windows	So
AGHILL	Wildflower	OS400	

Figure E-26 Association By Person report, first page



Associations By Person

01/14/2004
21:17:13

Person / User Identity	Description / Registry	Registry Type	Association Type
Zoe S. Torres	Account Representative		
ZST	Minnesota	OS400	Target
TORRESZ	NewYork	RACF	Target
ZTorres	Texas	AIX	Target
ZSTL1031A	Utah	NDS	Target
ZOETORRES	Washington	Windows	Source
ZSTORRE	Wildflower	OS400	Target

TOTAL PEOPLE: 205

Figure E-27 Association By Person report, last page

Summary

TIM SSO makes it easy for a security administrator to quickly and accurately create and maintain EIM associations between different user identities across the enterprise and a unique person.

In addition to an intuitive GUI designed exclusively for EIM management, TIM SSO provides wizards to simplify key functions. Wizards exist to import data for the people, registries, and user identities within an organization. A unique and powerful TIM SSO feature is its ability to identify suggested associations based on customizable user identity composition rules. Building on this capability, TIM SSO makes it easy and fast to add all the associations for one or more servers to a domain.

For more information, visit:

http://www.triaworks.com

F

Java code listings and output examples

In this appendix you will find the code listings and output as detailed in this redbook.

ReportEIM class

This code listing is the code listing for the ReportEIM class.

Example: F-1 ReportEIM class

```
import com.ibm.eim.*;
import java.util.*;

/**
 * Craig Wayman Bsc CLP
 * Any comments or suggestions you can contact me at
 * Craig_Wayman@Typex.com
 *
 * For a more in depth explanation of the methods and properties
 * used in this example please consult the IBM Redbook
 *
 * Java Code Example For EIM
 *
 * (C)Copyright IBM Corp.  2003, 2003
 *
 * The Source code for this program is not published  or otherwise
 * divested of its trade secrets,  irrespective of what has been
 * deposited with the U.S. Copyright Office.
 *
 * Provided as-is with no support, For education/example only.
 */

public class ReportEIM
{
   private DomainManager dm = DomainManager.getInstance();
   private Domain d = null;

   //Information to connect with
   private String user = "cn=administrator";
   private String password = "ldappw";
   private ConnectInfo connectInfo = new ConnectInfo(user, password);
   private String system = "ldap://as20.itsoroch.ibm.com:389/";
   private String domain = "ibm-eimdomainname=BlueNotesEIMDomain";
   private String ldapInfo = system + domain;
   private String domaindesc = "Created By Blue Notes EIM Administrator (BlueNotes.com)";

   //Registries To Add
   private String sysregistry = "My_iSeries";
   private String sysregistrytype = Registry.EIM_REGTYPE_OS400;
   private int sysregistrykind = Registry.EIM_SYSTEM_REGISTRY;

   private String appregistry = "LDAP";
   private String appregistrytype = Registry.EIM_REGTYPE_LDAP;
   private String appregistrydesc = "My LDAP Server";

   private String sysregistry2 = "My_AIX";
   private String sysregistrytype2 = Registry.EIM_REGTYPE_AIX;

   private String sysregistry3 = "My_RACF";
   private String sysregistrytype3 = Registry.EIM_REGTYPE_RACF;

     //People To Add
   private String eidname1 = "Craig Wayman";
   private String eiddesc1 = "EIM Code Author";
   private String eidname2 = "Charlie Brown";
   private String eiddesc2 = "Loves baseball";
   private String eidname3 = "Eric";
   private String eiddesc3 = "When Eric eats a banana..";
   private String eidname4 = "Clark Kent";
   private String eiddesc4 = "Better than Eric?";

   private HashMap h, rh;

   public static final int ReportOne = 1;
   public static final int ReportTwo = 2;
   public static final int ReportThree = 3;
   public static final int ReportFour = 4;
   public static final int DeleteDomain = 5;

   /**
    * Constructor for ReportEIM.
    */

   public ReportEIM()
   {
      super();
   }

    /*
     * Get the domain. If domain doesn't exist, try
     * creating the domain.
     */
    private Set getAllDomains() throws EimException
   {
      System.out.println("Getting all domains ...");
      try
      {
         Set s = dm.getDomains(system, connectInfo);
         return s;
      }
      catch (EimException e)
      {
         System.out.println("Unable to get domains");
             System.out.println(e.toString());
             return null;
        }
    }

   private Domain getDomain(String domainName, boolean create)
   {
      System.out.println("Getting domain ...");
      try
      {
         return dm.getDomain(system + domainName, connectInfo);
      }
      catch (EimException e)
      {
         System.out.println("Unable to get existing domain");
             System.out.println(e.toString());
             try{return (create ? createDomain() : null);}
             catch(EimException ex){ex.printStackTrace();}
        }
      return null;
   }
```

```
  /*
* Create the domain. If domain can't be created,
* throw an exception.
*/
private Domain createDomain() throws EimException
{
   System.out.println("Creating new domain");
   try
   {
      d = dm.createDomain(ldapInfo, connectInfo, domaindesc);
      return d;
   }
   catch (EimException e)
   {
      System.out.println("Unable to create domain");
           //throw e;
           return null;
   }
}

  /*
* Get registry. If registry can't be retrieved,
* try creating the registry.
*/
private Set getRegistries(Domain d, boolean create) throws EimException
{
   Set s = d.getRegistries();
   try
   {
      if(!s.isEmpty())
         return s;
      else
         return (create ? createRegistries(d) : s);
    }
   catch (EimException e)
   {
      System.out.println("Unable to get registry.");
      e.printStackTrace();
      return s;
   }
}

  /*
* Add registry. If registry can't be added,
* throw an exception.
*/
private Set createRegistries(Domain d) throws EimException
{
   try
   {
      System.out.println("Adding registries.");
      Registry reg;
      reg = d.addSystemRegistry(sysregistry, sysregistrytype, "iSeries Production Server", null);
      reg.addAlias(new RegistryAlias("91.92.94.94","UserDefinedInAPIs"));
      reg.addAlias(new RegistryAlias("Just some text","DNSHostName'"));
      reg = d.addApplicationRegistry(appregistry, appregistrytype, appregistrydesc, sysregistry);
      reg.addAlias(new RegistryAlias("91.92.94.95","UserDefinedInAPIs"));
      reg.addAlias(new RegistryAlias("hlp.me.with.these.apis","DNSHostName"));
      reg = d.addSystemRegistry(sysregistry2, sysregistrytype2, "AIX Box", null);
```

```
         reg.addAlias(new RegistryAlias("91.92.94.96","UserDefinedInAPIs"));
         reg = d.addSystemRegistry(sysregistry3, sysregistrytype3, "RACF Box", null);
         reg.addAlias(new RegistryAlias("91.92.94.97","UserDefinedInAPIs"));
         return d.getRegistries();
      }
      catch (EimException e)
      {
         System.out.println("Unable to add registry.");
         e.printStackTrace();
      }
      return null;
   }

     /*
   * Get eid. If eid can't be retrieved,
   * try adding the eid.
   */
   private Set getEids(Domain d, boolean create) throws EimException
   {
      Set s = d.getEids();
      try
      {
         if(!s.isEmpty())
            return s;
         else
            return (create ? createEids(d) : s);
      }
      catch (EimException e)
      {
         System.out.println("Unable to get eid.");
         e.printStackTrace();
      }
      return s;
   }

     /*
   * Add eid. If eid can't be added,
   * throw an exception.
   */
   private Set createEids(Domain d) throws EimException
   {
      try
      {
         System.out.println("Adding eid ..");
         Eid eid;
         eid = d.addEid(eidname1,eiddesc1);
            eid.addAssociation(Association.EIM_SOURCE,"My_iSeries","WaymanC");
            eid.addAssociation(Association.EIM_TARGET,"LDAP","craig");
            eid.addAssociation(Association.EIM_ADMIN,"My_AIX","namyawC");
            eid.addAssociation(Association.EIM_TARGET,"My_RACF","graic");
            eid.addAdditionalInfo("emailAddress:Craig_Wayman@Typex.com");
            eid.addAlias("Craig Michael Wayman");
            eid.addAlias("Craig M Wayman");
         eid = d.addEid(eidname2,eiddesc2);
            eid.addAssociation(Association.EIM_TARGET,"My_iSeries","BrownC");
            eid.addAssociation(Association.EIM_TARGET,"LDAP","charlie");
            eid.addAssociation(Association.EIM_SOURCE,"My_AIX","nworBC");
            eid.addAssociation(Association.EIM_TARGET,"My_RACF","eilrahc");
            eid.addAdditionalInfo("emailAddress:CharlieBrown@Snoopy.com");
            eid.addAlias("Charlie \"Legend\" Brown");
         eid = d.addEid(eidname3,eiddesc3);
            eid.addAssociation(Association.EIM_SOURCE,"My_iSeries","Eric");
            eid.addAssociation(Association.EIM_TARGET,"LDAP","eRiC");
            eid.addAssociation(Association.EIM_SOURCE,"My_AIX","cirE");
            eid.addAssociation(Association.EIM_ADMIN,"My_RACF","RiCe");
            eid.addAdditionalInfo("emailAddress:Eric@29AcaciaRoad.com");
            eid.addAlias("Bananaman");
         eid = d.addEid(eidname4,eiddesc4);
            eid.addAssociation(Association.EIM_ADMIN,"My_iSeries","KentC");
            eid.addAssociation(Association.EIM_TARGET,"LDAP","clark");
            eid.addAssociation(Association.EIM_SOURCE,"My_AIX","Ctnek");
            eid.addAssociation(Association.EIM_TARGET,"My_RACF","kralc");
            eid.addAdditionalInfo("emailAddress:CK@DailyPlanet.com");
            eid.addAdditionalInfo("emailAddress:Superman@SaveThePlanet.com");
            eid.addAlias("Superman");
         return d.getEids();
      }
      catch (EimException e)
      {
         System.out.println("Unable to add eid.");
         e.printStackTrace();
      }
      return null;
   }

    /*
     *  Create a hash map for easy look up of association types.
     */
   private void createAssociationTypeHashMap()
   {
      h = new HashMap();
      h.put(new Integer(Association.EIM_ADMIN), "admin");
      h.put(new Integer(Association.EIM_ALL_ASSOC), "all");
      h.put(new Integer(Association.EIM_SOURCE), "source");
      h.put(new Integer(Association.EIM_SOURCE_AND_TARGET),"source and target");
      h.put(new Integer(Association.EIM_TARGET), "target");
   }

   private void createRegistryTypeHashMap()
   {
      rh = new HashMap();
      rh.put(Registry.EIM_REGTYPE_AIX, "AIX");
      rh.put(Registry.EIM_REGTYPE_KERBEROS_EX, "Kerberos EX");
      rh.put(Registry.EIM_REGTYPE_KERBEROS_IG, "Kerberos IG");
      rh.put(Registry.EIM_REGTYPE_LDAP, "LDAP");
      rh.put(Registry.EIM_REGTYPE_NDS, "NDS");
      rh.put(Registry.EIM_REGTYPE_OS400, "OS400");
      rh.put(Registry.EIM_REGTYPE_POLICY_DIRECTOR, "Policy Director");
      rh.put(Registry.EIM_REGTYPE_RACF, "RACF");
      rh.put(Registry.EIM_REGTYPE_WIN2K, "WIN2K");
      rh.put(Registry.EIM_REGTYPE_X509, "X509");
      rh.put(new Integer(Registry.EIM_ALL_REGISTRIES), "All Registries");
      rh.put(new Integer(Registry.EIM_APPLICATION_REGISTRY), "Application Registry");
      rh.put(new Integer(Registry.EIM_SYSTEM_REGISTRY), "System Registry");
   }

   private void outputDomainInfo(Domain d, boolean recursive)
   {
      try
      {
         System.out.println("Domain Information for :" + d.getName());
         System.out.println("\tDescription: " + (d.getDescription()=="" ? "*No Description Available*" :
d.getDescription()));
         System.out.println("\tDistinguished Name: " + d.getDn());
         System.out.println("\tHost: " + d.getHost());
         System.out.println("\tName: " + d.getName());
         System.out.println("\tPort: " + d.getPort());
         System.out.println("\tURL: " + d.getUrl());
         System.out.println("\tisConected: " + d.isConnected());
         System.out.println("\tisSSL: " + d.isSSL());
         System.out.println("");

         if(recursive)
         {
            Set registries = getRegistries(d, false);
            outputRegistryInfo(registries, recursive, true);
         }
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving domain information");
         e.printStackTrace();
      }
   }

   private void outputDomainInfo(Set domains, boolean recursive, boolean showEIDs, boolean eidRecurse)
   {
      try
      {
         Iterator it = domains.iterator();
         while(it.hasNext())
         {
            Domain d = getDomain((String)it.next(),false);
            outputDomainInfo(d,recursive);
            if(showEIDs)
            {
               Set eids = getEids(d, false);
               outputEidInfo(eids, eidRecurse);
            }
         }
      }
      catch(EimException e)
      {
         e.printStackTrace();
      }
   }

   private void outputRegistryInfo(Registry reg, boolean fromDomainOutput, boolean recursive)
   {
      try
      {
         String tab;

         if(fromDomainOutput)
            tab = "\t\t";
         else
            tab = "\t";

         System.out.println((fromDomainOutput ? "\t" : "") + "Registry Information for: " +
reg.getName());
```

```
         System.out.println(tab + "Description: " + (reg.getDescription()=="" ? "*No Description 
Available*" : reg.getDescription()));
         System.out.println(tab + "Name: " + reg.getName());
         System.out.println(tab + "Type: " + reg.getType());
         System.out.println(tab + "Type Translated: " + rh.get(reg.getType()));
         System.out.println(tab + "Kind: " + reg.getKind());
         System.out.println(tab + "Kind Translated: " + rh.get(new Integer(reg.getKind())));
         System.out.println(tab + "Uuid: " + reg.getUuid());
         Set aliases = reg.getAliases();
         if(!aliases.isEmpty())
            outputRegistryAliasInformation(aliases, (fromDomainOutput) ? "\t\t" : "\t");

         switch(reg.getKind())
         {
            case Registry.EIM_SYSTEM_REGISTRY: //System Registry
               SystemRegistry sysReg = (SystemRegistry)reg;
               System.out.println(tab + "Uri: " + (sysReg.getUri()=="" ? "*No Uri Defined*" : 
sysReg.getUri()));
               break;

            case Registry.EIM_APPLICATION_REGISTRY: //Application Registry
               ApplicationRegistry appReg = (ApplicationRegistry)reg;
               System.out.println(tab + "SystemRegistry Name: " + appReg.getSystemRegistryName());
               break;
         }

         System.out.println("");

         if(recursive)
            outputRegistryUsersInfo(reg, recursive);
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving registry information");
         e.printStackTrace();
      }
   }

   private void outputRegistryInfo(Set reg, boolean fromDomainOutput, boolean recursive)
   {
      Iterator it = reg.iterator();
      while(it.hasNext())
         outputRegistryInfo((Registry)it.next(),fromDomainOutput,recursive);
   }

   private void outputEidInfo(Eid eid, boolean recurse)
   {
      try
      {
         System.out.println("Eid information for: " + eid.getName());
         System.out.println("\tName: " + eid.getName());
         System.out.println("\tDescription: " + eid.getDescription());
         System.out.println("\tUuid: " + eid.getUuid());
         Set aliases = eid.getAliases();
         if(!aliases.isEmpty())
            outputStringInformation(aliases, "Alias", "\t");
         Set additionalInfo = eid.getAdditionalInfo();
         if(!additionalInfo.isEmpty())
            outputStringInformation(additionalInfo, "Additional", "\t");
         System.out.println("");
         if(recurse)
            outputAssociationInfo(eid.getAssociations(Association.EIM_ALL_ASSOC), true);
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving eid information");
         e.printStackTrace();
      }
   }

   private void outputEidInfo(Set eids, boolean recurse)
   {
      Iterator it = eids.iterator();
      while(it.hasNext())
         outputEidInfo((Eid)it.next(), recurse);
   }

   private void outputAssociationInfo(Association assoc, boolean fromEidOutput)
   {
      try
      {
         String tab;

         if(fromEidOutput)
            tab = "\t\t";
         else
            tab = "\t";

         System.out.println((fromEidOutput ? "\t" : "") + "Association information for: " +
(assoc.getEid()).getName());
         System.out.println(tab + "Association Type: " + assoc.getAssociationType());
         System.out.println(tab + "Association Type Translation: " + h.get(new
Integer(assoc.getAssociationType())));
         System.out.println(tab + "Registry: " + assoc.getRegistryName());
         System.out.println(tab + "User ID: " + assoc.getUid());
         System.out.println("");
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving eid information");
         e.printStackTrace();
      }
   }

   private void outputAssociationInfo(Set assoc, boolean fromEidOutput)
   {
      Iterator it = assoc.iterator();
      while(it.hasNext())
         outputAssociationInfo((Association)it.next(), fromEidOutput);
   }

   private void outputRegistryUserInfo(RegistryUser regUsr, boolean fromRegistryOutput)
   {
      try
      {
         String tab;

         if(fromRegistryOutput)
            tab = "\t\t\t";
         else
            tab = "\t\t";

         System.out.println((fromRegistryOutput ? "\t\t" : "\t") + "RegistryUser Information for this 
registry");
         System.out.println(tab + "Description: " + (regUsr.getDescription()=="" ? "*No Description 
Available*" : regUsr.getDescription()));
         System.out.println(tab + "Registry Name: " + regUsr.getRegistryName());
         System.out.println(tab + "Target User Name: " + regUsr.getTargetUserName());
         System.out.println("");
      }
      catch (EimException e)
      {
         System.out.println("Error retrieving registry information");
         e.printStackTrace();
      }
   }

   private void outputRegistryUsersInfo(Registry reg, boolean fromRegistryOutput)
   {
      try
      {
         Iterator it = reg.getUsers().iterator();
         while(it.hasNext())
            outputRegistryUserInfo((RegistryUser)it.next(),fromRegistryOutput);
      }
      catch(EimException e)
      {
         System.out.println("Error accessing registry user");
         e.printStackTrace();
      }
   }

   private void outputStringInformation(Set info, String label, String tabs)
   {
      Iterator it = info.iterator();
      System.out.println(tabs + label + " information:");
      while(it.hasNext())
         System.out.println(tabs + "\t" + label + ": " + ((String)it.next()));
   }

   private void outputRegistryAliasInformation(RegistryAlias regAlias, String tabs)
   {
      System.out.println(tabs + "\t" + "Value: " + regAlias.getName() + "   Type: " + regAlias.getType());
   }

   private void outputRegistryAliasInformation(Set regAliases, String tabs)
   {
      Iterator it = regAliases.iterator();
      System.out.println(tabs + "Alias information:");
      while(it.hasNext())
         outputRegistryAliasInformation((RegistryAlias)it.next(),tabs);
   }

   private void deleteEimDomain(String domainName)
   {
      try
      {
         d = getDomain(domainName,false);
         if(d!=null)
```

```
        {
           Set eids = getEids(d, false);
           Iterator it = eids.iterator();
           while(it.hasNext())
           {
              Eid eid = ((Eid)it.next());
              eid.delete();
           }

           //get registries
           Registry temp;
           Set registries = getRegistries(d, false);
           it = registries.iterator();
           //Must first remove appl registries as reference system registries
           while(it.hasNext())
           {
              temp = (Registry)it.next();
              if(temp.getKind()==Registry.EIM_APPLICATION_REGISTRY)
                 temp.delete();
           }

           //Then remove the systems
           registries = getRegistries(d, false);
           it = registries.iterator();
           while(it.hasNext())
           {
              temp = (Registry)it.next();
              if(temp.getKind()==Registry.EIM_SYSTEM_REGISTRY)
                 temp.delete();
           }

           //In case there are any not defined as either type
           registries = getRegistries(d, false);
           it = registries.iterator();
           while(it.hasNext())
           {
              temp = (Registry)it.next();
              if(temp.getKind()==Registry.EIM_ALL_REGISTRIES)
                 temp.delete();
           }

           d.delete();
           System.out.println("Delete Successful");
        }
        else
           System.out.println("Canont retrieve domain object");
     }
     catch(EimException e)
     {
        System.out.println("Cannot complete delete successfully");
        e.printStackTrace();
     }
  }

  public void startReport(int reportToRun)
  {
     //Setup Hash Tables For Quick Lookup Of Constants
     createAssociationTypeHashMap();
     createRegistryTypeHashMap();
```

```
      switch(reportToRun)
      {
         case ReportOne:
            reportOne();
         break;

         case ReportTwo:
            reportTwo();
         break;

         case ReportThree:
            reportThree();
         break;

         case ReportFour:
            reportFour();
         break;

         case DeleteDomain:
            //Method used to delete all elements in a domain
            deleteEimDomain("ibm-eimdomainname=BlueNotesEIMDomain");
         break;
      }
   }

   private void reportOne()
   {
      try
      {
      /* This report first creates the information in EIM based on the default information
         Contained within this class and then creates a call which will display all information
         about the domain , its registries and the registry users for those registries*/
         Domain d = getDomain("ibm-eimdomainname=BlueNotesEIMDomain",true);
         Set registries = createRegistries(d);
         Set eids = createEids(d);
         outputDomainInfo(d, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }

   private void reportTwo()
   {
      try
      {
      /* This report retrieves an existing domain object and displays information about the domain only
         A list of registries is then retrieved and this information is outputted*/
         Domain d = getDomain("ibm-eimdomainname=BlueNotesEIMDomain",false);
         outputDomainInfo(d, false);
         Set registries = getRegistries(d, false);
         outputRegistryInfo(registries, false, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }

   private void reportThree()
   {
      try
      {
      /* This report accesses a domain object and outputs all the identifiers and
         their associations contained within the EIM domain*/
         Domain d = getDomain("ibm-eimdomainname=BlueNotesEIMDomain",false);
         Set eids = getEids(d, false);
         outputEidInfo(eids, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }

   private void reportFour()
   {
      try
      {
      /* This report retrieves a list of all the EIM domains contained in the LDAP server
         and outputs all information about the domains, registries, registry users and
         identifiers with their associations                    */
         Set domains = getAllDomains();
         outputDomainInfo(domains, true, true, true);
      }
      catch (EimException e)
      {
         System.out.println("Error Generating Report");
         System.out.println(e.toString());
      }
   }
}
```

Another class is needed to instantiate this object and call the startReport method. Here is an example of this class.

Example: F-2 The ReportAgent class

```
import ReportEIM;

public class ReportAgent
{
   public static void main(String args[])
   {
      System.out.println("Begin Report");
      ReportEIM rEIM = new ReportEIM();
//    rEIM.startReport(rEIM.ReportOne);
      rEIM.startReport(rEIM.ReportTwo);
//    rEIM.startReport(rEIM.ReportThree);
//    rEIM.startReport(rEIM.ReportFour);
//    rEIM.startReport(rEIM.DeleteDomain);
      System.out.println("End Report");
   }
}
```

Sample output for reportOne()

This is the sample output which is created when report one is created.

Example: F-3   ReportOne output

```
-- Begin Report --
Getting domain ...
Unable to get existing domain
EIM exception text=com.ibm.eim.jndi.DomainJNDI:connect: failed to connect to initial directory context.
Root exception class=class javax.naming.NameNotFoundException. Root exception text=[LDAP: error code 32 -
No Such Object].
Creating new domain
Adding registries.
Adding eid ..
Domain Information for :BlueNotesEIMDomain
   Description: Created By Blue Notes EIM Administrator (BlueNotes.com)
   Distinguished Name: ibm-eimdomainname=BlueNotesEIMDomain
   Host: as20.itsoroch.ibm.com
   Name: BlueNotesEIMDomain
   Port: 389
   URL: ldap://as20.itsoroch.ibm.com:389/ibm-eimdomainname=BlueNotesEIMDomain
   isConected: true
   isSSL: false

   Registry Information for: My_iSeries
      Description: iSeries Production Server
      Name: My_iSeries
      Type: 1.3.18.0.2.33.2-caseIgnore
      Type Translated: OS400
      Kind: 1
      Kind Translated: System Registry
      Uuid: 9c4e3000-895a-186e-8a4e-0004ac018327
      Alias information:
         Value: 91.92.94.94   Type: UserDefinedInAPIs
         Value: Just some text   Type: DNSHostName'
      Uri: *No Uri Defined*

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_iSeries
         Target User Name: BROWNC

      Registry Information for: My_AIX
      Description: AIX Box
      Name: My_AIX
      Type: 1.3.18.0.2.33.5-caseExact
      Type Translated: AIX
      Kind: 1
      Kind Translated: System Registry
      Uuid: c860c000-895a-186e-8a4e-0004ac018327
      Alias information:
         Value: 91.92.94.96   Type: UserDefinedInAPIs
      Uri: *No Uri Defined*

   Registry Information for: My_RACF
      Description: RACF Box
      Name: My_RACF
      Type: 1.3.18.0.2.33.1-caseIgnore
      Type Translated: RACF
      Kind: 1
      Kind Translated: System Registry
      Uuid: d8508000-895a-186e-8a4e-0004ac018327
      Alias information:
         Value: 91.92.94.97   Type: UserDefinedInAPIs
      Uri: *No Uri Defined*

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_RACF
         Target User Name: GRAIC

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_RACF
         Target User Name: KRALC

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_RACF
         Target User Name: EILRAHC

   Registry Information for: LDAP
      Description: My LDAP Server
      Name: LDAP
      Type: 1.3.18.0.2.33.7-caseIgnore
      Type Translated: LDAP
      Kind: 2
      Kind Translated: Application Registry
      Uuid: b3ee7800-895a-186e-8a4e-0004ac018327
      Alias information:
         Value: hlp.me.with.these.apis   Type: DNSHostName
         Value: 91.92.94.95   Type: UserDefinedInAPIs
      SystemRegistry Name: My_iSeries

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: CRAIG

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: CHARLIE

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: CLARK

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: ERIC
-- End Report --
```

Sample output for reportTwo()

This is the sample output which is created when report two is created.

Example: F-4 ReportTwo output

```
-- Begin Report --
Getting domain ...
Domain Information for :BlueNotesEIMDomain
   Description: Created By Blue Notes EIM Administrator (BlueNotes.com)
   Distinguished Name: ibm-eimdomainname=BlueNotesEIMDomain
   Host: as20.itsoroch.ibm.com
   Name: BlueNotesEIMDomain
   Port: 389
   URL: ldap://as20.itsoroch.ibm.com:389/ibm-eimdomainname=BlueNotesEIMDomain
   isConected: true
   isSSL: false

Registry Information for: My_iSeries
   Description: iSeries Production Server
   Name: My_iSeries
   Type: 1.3.18.0.2.33.2-caseIgnore
   Type Translated: OS400
   Kind: 1
   Kind Translated: System Registry
   Uuid: 9c4e3000-895a-186e-8a4e-0004ac018327
   Alias information:
      Value: 91.92.94.94   Type: UserDefinedInAPIs
      Value: Just some text   Type: DNSHostName'
   Uri: *No Uri Defined*

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_iSeries
         Target User Name: BROWNC

Registry Information for: My_AIX
   Description: AIX Box
   Name: My_AIX
   Type: 1.3.18.0.2.33.5-caseExact
   Type Translated: AIX
   Kind: 1
   Kind Translated: System Registry
   Uuid: c860c000-895a-186e-8a4e-0004ac018327
   Alias information:
      Value: 91.92.94.96   Type: UserDefinedInAPIs
   Uri: *No Uri Defined*

Registry Information for: My_RACF
   Description: RACF Box
   Name: My_RACF
   Type: 1.3.18.0.2.33.1-caseIgnore
   Type Translated: RACF
   Kind: 1
   Kind Translated: System Registry
   Uuid: d8508000-895a-186e-8a4e-0004ac018327
   Alias information:
      Value: 91.92.94.97   Type: UserDefinedInAPIs
   Uri: *No Uri Defined*
```

```
      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_RACF
         Target User Name: GRAIC

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_RACF
         Target User Name: KRALC

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: My_RACF
         Target User Name: EILRAHC

Registry Information for: LDAP
   Description: My LDAP Server
   Name: LDAP
   Type: 1.3.18.0.2.33.7-caseIgnore
   Type Translated: LDAP
   Kind: 2
   Kind Translated: Application Registry
   Uuid: b3ee7800-895a-186e-8a4e-0004ac018327
   Alias information:
      Value: hlp.me.with.these.apis   Type: DNSHostName
      Value: 91.92.94.95   Type: UserDefinedInAPIs
   SystemRegistry Name: My_iSeries

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: CRAIG

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: CHARLIE

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: CLARK

      RegistryUser Information for this registry
         Description: *No Description Available*
         Registry Name: LDAP
         Target User Name: ERIC
-- End Report --
```

Sample output for reportThree()

This is the sample output which is created when report three is created.

Example: F-5 ReportThree output

```
-- Begin Report --
Getting domain ...
Eid information for: Craig Wayman
```

```
   Name: Craig Wayman
   Description: EIM Code Author
   Uuid: ea557000-895a-186e-8a4e-0004ac018327
   Alias information:
      Alias: Craig Wayman
      Alias: Craig Michael Wayman
      Alias: Craig M Wayman
   Additional information:
      Additional: emailAddress:Craig_Wayman@Typex.com

   Association information for: Craig Wayman
      Association Type: 1
      Association Type Translation: target
      Registry: LDAP
      User ID: CRAIG

   Association information for: Craig Wayman
      Association Type: 1
      Association Type Translation: target
      Registry: My_RACF
      User ID: GRAIC

   Association information for: Craig Wayman
      Association Type: 2
      Association Type Translation: source
      Registry: My_iSeries
      User ID: WAYMANC

   Association information for: Craig Wayman
      Association Type: 4
      Association Type Translation: admin
      Registry: My_AIX
      User ID: namyawC

Eid information for: Clark Kent
   Name: Clark Kent
   Description: Better than Eric?
   Uuid: 55f67000-895b-186e-8a4e-0004ac018327
   Alias information:
      Alias: Superman
      Alias: Clark Kent
   Additional information:
      Additional: emailAddress:CK@DailyPlanet.com
      Additional: emailAddress:Superman@SaveThePlanet.com

   Association information for: Clark Kent
      Association Type: 2
      Association Type Translation: source
      Registry: My_AIX
      User ID: Ctnek

   Association information for: Clark Kent
      Association Type: 1
      Association Type Translation: target
      Registry: My_RACF
      User ID: KRALC

   Association information for: Clark Kent
      Association Type: 1
      Association Type Translation: target
      Registry: LDAP
      User ID: CLARK

   Association information for: Clark Kent
      Association Type: 4
      Association Type Translation: admin
      Registry: My_iSeries
      User ID: KENTC

Eid information for: Charlie Brown
   Name: Charlie Brown
   Description: Loves baseball
   Uuid: 1940a800-895b-186e-8a4e-0004ac018327
   Alias information:
      Alias: Charlie "Legend" Brown
      Alias: Charlie Brown
   Additional information:
      Additional: emailAddress:CharlieBrown@Snoopy.com

   Association information for: Charlie Brown
      Association Type: 1
      Association Type Translation: target
      Registry: LDAP
      User ID: CHARLIE

   Association information for: Charlie Brown
      Association Type: 1
      Association Type Translation: target
      Registry: My_iSeries
      User ID: BROWNC

   Association information for: Charlie Brown
      Association Type: 1
      Association Type Translation: target
      Registry: My_RACF
      User ID: EILRAHC

   Association information for: Charlie Brown
      Association Type: 2
      Association Type Translation: source
      Registry: My_AIX
      User ID: nworBC

Eid information for: Eric
   Name: Eric
   Description: When Eric eats a banana..
   Uuid: 3b2d9800-895b-186e-8a4e-0004ac018327
   Alias information:
      Alias: Bananaman
      Alias: Eric
   Additional information:
      Additional: emailAddress:Eric@29AcaciaRoad.com

   Association information for: Eric
      Association Type: 1
      Association Type Translation: target
      Registry: LDAP
      User ID: ERIC

   Association information for: Eric
```

```
      Association Type: 2
      Association Type Translation: source
      Registry: My_AIX
      User ID: cirE

   Association information for: Eric
      Association Type: 2
      Association Type Translation: source
      Registry: My_iSeries
      User ID: ERIC

   Association information for: Eric
      Association Type: 4
      Association Type Translation: admin
      Registry: My_RACF
      User ID: RICE
-- End Report --
```

Sample output for reportFour()

This is the sample output which is created when report four is created. I have only listed the domain information here as the registry, registry user and identifier information is the same as is listed in the previous examples.

Example: F-6 ReportFour output

```
-- Begin Report --
Getting all domains ...
Getting domain ...
Domain Information for :ITSO EIM
   Description: Created by wizard.
   Distinguished Name: ibm-eimdomainname=itso eim
   Host: as20.itsoroch.ibm.com
   Name: ITSO EIM
   Port: 389
   URL: ldap://as20.itsoroch.ibm.com:389/ibm-eimdomainname=itso eim
   isConected: true
   isSSL: false

   It will then display all registries and user registries
   It will then display all identifiers

Domain Information for :BlueNotesEIMDomain
   Description: Created By Blue Notes EIM Administrator (BlueNotes.com)
   Distinguished Name: ibm-eimdomainname=bluenoteseimdomain
   Host: as20.itsoroch.ibm.com
   Name: BlueNotesEIMDomain
   Port: 389
   URL: ldap://as20.itsoroch.ibm.com:389/ibm-eimdomainname=bluenoteseimdomain
   isConected: true
   isSSL: false

   It will then display all registries and user registries
   It will then display all identifiers
```

EIMAuthorities class

This is the code listing for the EIMAuthorities class:

Example: F-7 EIMAuthorities class

```
import com.ibm.eim.*;
import java.util.*;
import java.io.*;

public class EIMAuthorities
{
   DomainManager dm;
   HashMap ah;

   public EIMAuthorities()
   {
      dm = DomainManager.getInstance();
   }
   public void createReport()
   {
      try
      {
         String ldapInfo = "ldap://as20.itsoroch.ibm.com:389/ibm-eimdomainname=BlueNotesEIMDomain";
         ConnectInfo ci = new ConnectInfo("cn=administrator","ldappw");
         Domain d = dm.getDomain(ldapInfo, ci);
         AccessContext ac = d.getAccessContext();

         createEIMAuthoritiesHashmap();

         System.out.println("EIM Authority Information:");
         System.out.println("");

         Set temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_ADMIN);
         outputStringInformation(temp, "Admin User", "\t");

         temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_IDENTIFIER_ADMIN);
         outputStringInformation(temp, "Identifier Admin User", "\t");

         temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_MAPPING_LOOKUP);
         outputStringInformation(temp, "Mapping Lookup User", "\t");

         temp = ac.getAdminAccessUsers(AccessContext.EIM_ACCESS_REG_ADMIN);
         outputStringInformation(temp, "Registry Admin User", "\t");

         System.out.println("");
         System.out.println("EIM Registry Authority Information:");
         System.out.println("");

         temp = ac.getRegistryAccessUsers("My_iSeries");
         outputStringInformation(temp, "My_iSeries Registry Admin User", "\t");

         temp = ac.getRegistryAccessUsers("My_RACF");
         outputStringInformation(temp, "My_RACF Registry Admin User", "\t");

         System.out.println("");
         System.out.println("User Context Information");
         System.out.println("");
         UserAccess ua =
ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"os400-profile=itscid12,cn=accounts,os400-sys=as20.itso.ibm.co
m");
         outputUserAccessInformation(ua, "itscid12", "\t");

         ua =
ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.co
m");
         outputUserAccessInformation(ua, "itscid11", "\t");

         ua =
ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"os400-profile=itscid10,cn=accounts,os400-sys=as20.itso.ibm.co
m");
         outputUserAccessInformation(ua, "itscid10", "\t");

         ua = ac.getUserAccess(AccessContext.EIM_ACCESS_KERBEROS,"krbsvr400");
         outputUserAccessInformation(ua, "krbsvr400", "\t");

         ua = ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"ibm-kn=krbsvr400");
         outputUserAccessInformation(ua, "krbsvr400", "\t");

         ua = ac.getUserAccess(AccessContext.EIM_ACCESS_DN,"cn=administrator");
         outputUserAccessInformation(ua, "cn=administrator", "\t");
      }
      catch(EimException e)
      {
         e.printStackTrace();
      }
   }

   public void createEIMAuthoritiesHashmap()
   {
      ah = new HashMap();
      ah.put(new Integer(AccessContext.EIM_ACCESS_ADMIN),"Administrative Access");
      ah.put(new Integer(AccessContext.EIM_ACCESS_DN),"User Type For LDAP In Distinguished Name Format");
      ah.put(new Integer(AccessContext.EIM_ACCESS_IDENTIFIER_ADMIN),"Identifier Access");
      ah.put(new Integer(AccessContext.EIM_ACCESS_KERBEROS),"User Type For LDAP In Kerberos Principal
Format");
      ah.put(new Integer(AccessContext.EIM_ACCESS_MAPPING_LOOKUP),"Mapping Lookup Access");
      ah.put(new Integer(AccessContext.EIM_ACCESS_REG_ADMIN),"All Registry Access");
   }

   private void outputStringInformation(Set info, String label, String tabs)
   {
      Iterator it = info.iterator();
      System.out.println(tabs + label + " information:");
      while(it.hasNext())
         System.out.println(tabs + "\t" + label + ": " + ((String)it.next()));
   }

   private void outputUserAccessInformation(UserAccess ua, String name, String tabs)
   {
      System.out.println("Output User Access Information for " + name);
      System.out.println(tabs + "\t" + "Dn: " + ua.getDn());
      Set temp = ua.getRegistries();
      if(!temp.isEmpty())
         outputStringInformation(temp, "Admin authority to following Registries", tabs + "\t");
      System.out.println(tabs + "\t" + "hasAdmin: " + ua.hasAdmin());
      System.out.println(tabs + "\t" + "hasEidAdmin: " + ua.hasEidAdmin());
      System.out.println(tabs + "\t" + "hasRegistryAdmin: " + ua.hasRegistryAdmin());
      System.out.println(tabs + "\t" + "hasMappingLookup: " + ua.hasMappingLookup());
      System.out.println("");
   }
}
```

This class is instantiated and the createReport method called. Here is a sample class on how to perform this function:

Example: F-8 Example class for EIMAuthorities

```
import EIMAuthorities;

public class ReportAgent
{
   public static void main(String args[])
   {
      System.out.println("Begin Report");
      EIMAuthorities eimAuth = new EIMAuthorities();
      eimAuth.createReport();
      System.out.println("End Report");
   }
}
```

EIMAuthorities output

This is the output which is created from the createReport method of the EIMAuthorities class:

Example: F-9 EIMAuthorities output

```
-- Begin Report --
EIM Authority Information:

   Admin User information:
      Admin User: os400-profile=itscid12,cn=accounts,os400-sys=as20.itso.ibm.com
      Admin User: cn=administrator
      Admin User: ibm-eimdomainname=bluenoteseimdomain

   Identifier Admin User information:
      Identifier Admin User: ibm-kn=krbsvr400
      Identifier Admin User: ibm-eimdomainname=bluenoteseimdomain

   Mapping Lookup User information:
      Mapping Lookup User: os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.com
      Mapping Lookup User: ibm-kn=krbsvr400
      Mapping Lookup User: ibm-eimdomainname=bluenoteseimdomain

   Registry Admin User information:
      Registry Admin User: os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.com
      Registry Admin User: ibm-eimdomainname=bluenoteseimdomain

EIM Registry Authority Information:

   My_iSeries Registry Admin User information:
      My_iSeries Registry Admin User: os400-profile=itscid10,cn=accounts,os400-sys=as20.itso.ibm.com
      My_iSeries Registry Admin User: ibm-eimdomainname=bluenoteseimdomain

   My_RACF Registry Admin User information:
      My_RACF Registry Admin User: ibm-eimdomainname=bluenoteseimdomain

User Context Information

Output User Access Information for itscid12
      Dn: os400-profile=itscid12,cn=accounts,os400-sys=as20.itso.ibm.com
      hasAdmin: true
      hasEidAdmin: false
      hasRegistryAdmin: false
      hasMappingLookup: false

Output User Access Information for itscid11
      Dn: os400-profile=itscid11,cn=accounts,os400-sys=as20.itso.ibm.com
      hasAdmin: false
      hasEidAdmin: false
      hasRegistryAdmin: true
      hasMappingLookup: true

Output User Access Information for itscid10
      Dn: os400-profile=itscid10,cn=accounts,os400-sys=as20.itso.ibm.com
      Admin authority to following Registries information:
         Admin authority to following Registries: LDAP
         Admin authority to following Registries: My_iSeries
         Admin authority to following Registries: My_AIX
      hasAdmin: false
      hasEidAdmin: false
      hasRegistryAdmin: false
      hasMappingLookup: false

Output User Access Information for krbsvr400
      Dn: ibm-kn=krbsvr400
      hasAdmin: false
      hasEidAdmin: true
      hasRegistryAdmin: false
      hasMappingLookup: true

Output User Access Information for cn=administrator
      Dn: cn=administrator
      hasAdmin: true
      hasEidAdmin: false
      hasRegistryAdmin: false
      hasMappingLookup: false
-- End Report --
```

G

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246975

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG24-6975.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

File name	*Description*
DominoWebAccess.zip	Contains a SAVF with the source code for the Domino Web Access scenario.
EIMDemoTool.zip	Contains the source code for the EIM Demo Tool and a ReadMe file for installation instructions.
WebSphereScenario.zip	Provides the Java files (source) and RPG programs used in the WebSphere scenario.

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	1.5 Megabytes (if all three file are downloaded)
Operating system:	All scenarios are based on Windows 2000 or XP client systems

Processor:	No specific requirement for download
Memory:	No specific requirement for download

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder. Follow the instructions in the section related to the downloaded material or the readme file associated with the file.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 296. Note that some of the documents referenced here may be available in softcopy only.

- *Getting the Most from Your Domino Directory,* SG24-5986
- *Using LDAP for Directory Integration*, SG24-6163
- *Implementation and Practical Use of LDAP on the IBM @server iSeries Server*, SG24-6193
- *Integrating Lotus Domino 6 and WebSphere Express V5 on the IBM @server iSeries Server*, SG24-6998
- *LDAP Implementation Cookbook*, SG24-5110
- *IBM @server iSeries IP Networks: Dynamic!*, SG24-6718
- *iSeries Access for Web V5R2 and WebSphere Host Publisher V4.0*, SG24-6804
- *Understanding LDAP*, SG24-4986

Other publications

These publications are also relevant as further information sources:

- *Kerberos, A Network Authentication System*, by Brian Tung. Addison-Wesley Publishing Company, May 1999. ISBN: 0-201-37924-4
- *Kerberos, The Definitive Guide*, by Jason Garman. O’Reilly and Assoociates, August 2003. ISBN: 0-596-00403-6
- *iSeries Access for Web V5R2*, SC41-5518
- *Backup and Recovery V5R2*, SC41-5304

Online resources

These Web sites and URLs are also relevant as further information sources:

- iSeries Information Center for Release 5:

  http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm
- IBM @server Enterprise Identity Mapping Web page:

  http://www-1.ibm.com/servers/eserver/security/eim/
- Massachusetts Institute of Technology Kerberos home page:

  http://web.mit.edu/kerberos/dist/index.html

- Internet Requests For Comments (RFC) index page:

  http://www.faqs.org/rfcs/
- iSeries Access home page:

  http://www-1.ibm.com/servers/eserver/iseries/access/index.html
- Lotus documentation:

  http://www-10.lotus.com/ldd
- Search specific questions about Microsoft implementation of SSO or Windows at:

  http://www.microsoft.com
- Typex's BlueNotes:

  http://www.bluenotes.com
- SafeStone's AxcessIT:

  http://www.safestone.com
- WebSphere Host On-Demand Version 8 home page:

  http://www-306.ibm.com/software/webservers/hostondemand/library/v8infocenter/hod/en/help/2tabcontents.html

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Redbooks

Windows-based Single Signon and the EIM Framework on the IBM @server iSeries Server

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

Windows-based Single Signon and the EIM Framework on the IBM eserver iSeries Server

Configure your iSeries to participate in a single signon environment

Network Authentication Service on your network and the iSeries

Enterprise Identity Mapping explained

Support for a Kerberos based Network Authentication Service and the introduction of Enterprise Identity Mapping (EIM) were exciting OS/400® V5R2 announcements during 2002.

A Kerberos based Network Authentication Service enables the iSeries (and any kerberized application) to use a Kerberos ticket for authentication instead of a user ID and password. This enables you to sign on once in the morning to your Kerberos based security server and not be prompted again when accessing your enabled applications. This is called Single Signon (SSO).

Enterprise Identity Mapping (EIM) is a cross platform solution that involves a wide range of technologies including Kerberos, LDAP, and Kerberos Network Authentication Service. Basically, EIM is a framework provided by IBM that allows the mapping of authenticated users to OS/400 (and application) userids. This extends the power of SSO to the enterprise.

Because the iSeries is well known as a server that can consolidate a wide range of application programming environments into one manageable system, this IBM Redbook, then, studies the implementation of Kerberos and EIM in a SCON environment that includes OS/400, Windows, and applications that are right now being updated to support the new framework. We provide easy to follow examples that demonstrate all the pieces working together.

SG24-6975-00 ISBN 0738498998